Proceedings of the 10th International Conference on Bioinformatics and Computational Biology (BICOB 2018)

March 19–21, 2018

Las Vegas, Nevada, USA

Editors: Hisham Al-Mubaid, Qin Ding, Oliver Eulenstein

ISBN: 978–1–943436–11–8

Printed in Winona, MN, USA — March 2018

International Program Committee

Conference Chair

Oliver Eulenstein	Iowa State University, USA

Program Chairs

Hisham Al-Mubaid	University of Houston – Clear Lake, USA
Qin Ding	East Carolina University, USA

Publicity Chair

Nurit Haspel	University of Massachusetts Boston, USA

Committee Members

Abdullah Arslan	Texas A & M University - Commerce, USA
Mukul S. Bansal	University of Connecticut, USA
Paola Bonizzoni	Universita di Milano-Bicocca, Italy
Daniel Brown	University of Waterloo, Canada
Kun-Mao Chao	National Taiwan University, Taiwan
Lin Chen	Elizabeth City State University, USA
Ping Chen	University of Massachusetts Boston, USA
Jianlin Cheng	University of Missouri Columbia, USA
Peter Clote	Boston College, USA
Scott Emrich	University of Notre Dame, USA
Krishnendu Ghosh	Miami University, USA
Paweł Górecki	University of Warsaw, Poland
Osamu Gotoh	National Institute of Advanced Industrial Science and Technology (AIST), USA
Matthew Hayes	Xavier University of Louisiana, USA
Filip Jagodzinski	Western Washington University, USA
Danny Krizanc	Wesleyan University, USA
Keith Marsolo	Cincinnati Children's Hospital Medical Center, USA
Bernard Moret	Ecole Polytechnique Federale de Lausanne, Switzerland
Kamal Al Nasr	Tennessee State University, USA
Tayo Obafemi-Ajayi	Missouri State University, USA
Aida Ouangraoua	Universite de Sherbrooke, Canada
Yann Ponty	CNRS/LIX, Polytechnique, France
Jianhua Ruan	The University of Texas at San Antonio, USA
Ugo Vaccaro	University of Salerno, Italy
Jianxin Wang	Central South University, China
Li-San Wang	University of Pennsylvania, USA
Bartek Wilczynski	University of Warsaw, Poland
Ka-Chun Wong	City University of Hong Kong, Hong Kong
Jin Zhang	Washington University in St. Louis, USA
Louxin Zhang	National University of Singapore, Singapore
Jaroslaw Zola	University at Buffalo, USA

Message from Chairs

One of the greatest values of any scientific conference is in the gathering of a large number of fellow researchers in the same place and same time. This is the main motivation and purpose of any conference, and this BICOB 2018 is no exception.

Having said that, we, at the organizing committee, are pleased to welcome you to the 10th International Conference on Bioinformatics and Computational Biology (BICOB 2018), sponsored by the International Society for Computers and Their Applications (ISCA). We also would like to express our gratitude for you taking the effort to come and present your research findings to us. This year, BICOB 2018 is held in Las Vegas, Nevada, USA, during March 19–21, 2018, seemingly a popular time and place for this conference.

BICOB is instituting in itself one of the important outlets of research presentations and discussions in the fields of bioinformatics and computational biology. The tenth BICOB conference offers the opportunity and utmost venue for researchers and scientists from all over the world to present and discuss their research results, techniques and findings with other researchers having similar interests in the field of bioinformatics. Also, BICOB encourages interesting applications and emerging challenges in the field.

Being considered as one of the best destinations for vacations, entertainment, and conventions in the world, Las Vegas was selected as the venue for BICOB 2018 which is held during March 19–21, 2018 in the heart of the entertainment capital of the world.

This time the conference features a keynote talk by Dr. Dan Gusfield, who is well known for his outstanding work in bioinformatics and computational biology. Dr. Gusfield is a distinguished professor coming from UC-Davis, California, USA, to tell us about his outstanding research results and findings in bioinformatics algorithm and biomedical combinatorics. This year, the main conference includes eight sessions of regular paper presentations. The conference addresses a broad range of central topics in the bioinformatics area including: next generation sequencing, biological networks, regulatory networks, genomics, proteomics, protein structure analysis, data mining and machine learning applications in bioinformatics, diseases and drug related research, microarray research and applications and more.

The conference is gathering bioinformatics researchers, scientists, practitioners, and attendants from many countries. Furthermore, the participants are coming from various research institutions like universities, corporations, and government research agencies.

In BICOB 2018, each submission was evaluated and peer-reviewed by three to five reviewers who are usually members of the international program committee (PC). The members of the PC with their affiliations and countries are listed in the proceedings and on the conference website. Papers have been evaluated by referees judging the originality, significance, technical contents, application contents and presentation style. We used the EasyChair online conference management system to automate the workflow for submission and refereeing.

Finally, we would like to gratefully acknowledge the professional work of the program committee and the sub-reviewers for contributing a tremendous amount of time. We owe great thanks to ISCA president and board of directors for the well-organized conference management. We also would like to thank the ISCA executive director for his help, guidance, support, and work. We also want to thank all presenters and attendees for actively contributing to the success of BICOB 2018 and look forward to excellent presentations and fruitful discussions at the meeting, which will certainly broaden our professional horizons. All participants are invited to make new friends within the BICOB family. We sincerely wish to every participant a very enjoyable and beneficial time at BICOB 2018. And as a final note: next year BICOB will be held in Honolulu, HI, USA (another great destination for meetings and entertainments; thus we invite you to consider it for your research presentation and meeting next year in March).

With Our Best Regards

Hisham Al-Mubaid, Program co-Chair
Qin Ding, Program co-Chair
Oliver Eulenstein, Conference Chair

March 2018

Table of Contents

Integer Linear Programming in Computational and Systems Biology

Keynote Address

Dan Gusfield
Department of Computer Science
University of California, Davis
Davis, CA 95616, USA
gusfield@cs.ucdavis.edu

Abstract

Integer (Linear) Programming, abbreviated "ILP", is a versatile modeling and optimization technique that was developed for complex planning and operational decision making. However, it has been increasingly used in computational biology in non-traditional ways, most importantly and inventively as a computational tool to model biological phenomena, to analyze biological data, and to extract biological insight from the models and the data. Integer programming is often very effective in solving instances of hard biological problems on realistic data of current importance, despite the fact that many of those problems lack general algorithmic solutions that are efficient (in a provable, worst-case sense), and that the problem of solving integer programs also lacks a provable worst-case efficient general solution.

Highly engineered, commercial ILP solvers are available (now free to academics and researchers) to solve ILP formulations. The improvement of the best solvers has been spectacular, with an estimate that (combined with faster computers) benchmark ILP problems can now be solved 200-billion times faster than twenty-five years ago. Exploiting ILP, some biological problems of importance can be modeled in a way that allows a solution in seconds on a laptop, while more common (statistically-based) models require days, weeks or months of computation on large clusters.

The effectiveness of the best ILP solvers on problem instances of importance in biology opens huge opportunities. The impact of faster and easier-to-implement computation could be truly transformative in several parts of biology. However, there are challenges in effectively using these tools for biological problems, and educational and outreach issues that must be addressed. In this talk, I will discuss some of the successes, opportunities, and challenges in exploiting ILP for computational and systems biology.

Bibliography

Dr. Gusfield's primary interests involve the efficiency of algorithms, particularly for problems in combinatorial optimization and graph theory. These algorithms have been applied to study data security, stable matching, network flow, matroid optimization, string/pattern matching problems, molecular sequence analysis, and optimization problems in population-scale genomics. Currently, he is focused on string and combinatorial problems that arise in computational biology and bioinformatics. Dr. Gusfield served as chair of the computer science department at UCD from July 2000 until August 2004, and was the founding Editor-in-Chief of The IEEE/ACM Transactions of Computational Biology and Bioinformatics until January 2009.

Dan Gusfield received his Ph.D. in 1980 from UC Berkeley, working with Richard Karp, and was an Assistant Professor at Yale University from 1980 to 1986. His dissertation concerned problems of

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

sensitivity analysis in graphs, network flow and Matroid theory. In January 1987 Dan moved to UC Davis. In July 2016, he was promoted to the rank of Distinguished Professor.

RepCalc: a Tool For Calculating Transposable Element Density within the Genome

Tamer Aldwairi
Computer Science and Engineering
Mississippi State University
Mississippi State, MS 39762, USA
taa70email@gmail.com

Federico Hoffmann
Biochemistry and Molecular Biology
Mississippi State University
Mississippi State, MS 39762, USA
fgh19@msstate.edu

Benjamin Elam
Mathematics and Statistics
Mississippi State University
Mississippi State, MS 39762, USA
ben.a.elam@gmail.edu

Andy D. Perkins
Computer Science and Engineering
Mississippi State University
Mississippi State, MS 39762, USA
perkins@cse.msstate.edu

Abstract

Transposable elements (TEs) are mobile genetic elements that comprise a large portion of the genome of many eukaryotic organisms. They can transpose directly through the use of cut/paste mechanisms or indirectly copy/paste mechanisms. TE's can sometimes induce a harmful effect through inserting themselves in a gene, rendering it non-functional. A number of tools have been developed with the goal of providing annotation information for the location of TEs within the genome. Our tool provides the researchers with ways to speed up the process of analyzing, comparing and summarizing the important information, in regards to the distribution of the densities for different families and subfamilies of TEs, whether across the whole genome or within specific regions of interest (ROI). Equipping researchers with such a tool is an imperative necessity to the research community. To demonstrate the usefulness of our tool we used Piwi-interacting RNA (piRNA) clusters as our ROI in which we calculate the densities for the different families/subfamilies of TEs. We observed that the densities of TEs varies and heavily depends on the location being analyzed, whether it is a piRNA cluster, a gene, or other parts of the genome.

1. Background And Motivation

TEs are one of the several types of mobile genetic elements, which can be defined as DNA sequences that can move from one position to another within the genome through replication and insertion. They can be divided into two main classes based on their transposition mechanism. The first type is retrotransposons, which use copy and paste mechanisms and move indirectly via an RNA intermediate. These include long terminal repeats (LTRs), long interspersed elements (LINEs) and short interspersed elements (SINEs). The second type is DNA transposons, which move directly through the use of cut and paste mechanisms. Retrotransposons are more abundant in plants as well as many other organisms than DNA transposons. For example, TEs account for approximately 45% of the human genome, 41%-42% are retrotransposons 20% are LINEs, 13% are SINEs, 8% are LTRs, while DNA transposons account for only 3% of the TEs within the human genome [1][2][3].

TEs have the potential of causing harm or damage to the host cell through their continuous movement within the genome (insertion, deletion, duplication). For example, they can insert themselves into a functional gene which could block or disable the functionality of that gene. However, most TEs are in a non-active state, meaning they are not duplicating or moving from one place to another in the genome. Even though active TEs might be potentially harmful, the genome has developed mechanisms to suppress and silence their activity [4][5]. Generally speaking, TEs can have a positive or a negative impact on the organization of an organism's genome and its progeny [4].

TEs have been an important topic of study since their discovery in the 1940s by McClintok [6]. Advancement of next-generation sequencing technologies and the large amount of data produced by these methods has led to the generation of many tools for the purpose of annotation and classification of TEs. Some of these tools are tailored specifically for TE identification while others are general purpose tools that incorporate identification of TEs as part of their overall framework. There is a numerous list of tools for annotation of TE, with Bergman [7] listing more than thirty tools while Lerat [8] listed more than fifty.

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

These tools include Repeat Masker [9] , CENSOR [10], RepeatFinder [11], RepSeek [12], DAWGPAWS [13], RepeatScout [14] and various others. Some tools like Repeat Masker and CENSOR enable the user to search for repeats by comparing them to a reference sequence within certain databases like Repbase [15], which is a repetitive element database for eukaryotic organisms. These tools differ significantly in their goals and the underlying mechanisms of how they work. Some of these are general purpose tools, while others perform specific tasks such as TE annotation or classification, and sometimes both tasks. What these tools have in common is that they provide the researcher with basic information on the annotation of TEs. However, we sometimes want to know the distribution and quantitative information for specific families and subfamilies of TE classes and whether these classes are abundant within certain regions [16] [17] [18] [19] [20] [21]. The tool described here allows researchers to easily compare the different regions within the genome to determine whether those regions share certain common characteristics with respect to the presence or absence of certain families or sub-families of TEs. It can also be used to highlight regions of high recombination rates through identifying those regions with high transposes and no retrotransposons [25]. It is important to point out that the tool cannot identify transposon-free regions [26] directly but those regions can be inferred from the results based on which regions have been identified to contain TEs. Here, we provide the researchers with a tool that automates the process of classifying the transposable elements into their main class of families and sub-families across different regions of interest and for the whole genome. Our tool automates, facilitates and simplifies this long and difficult task which is usually performed in a non-automated fashion. We chose to demonstrate the functionality of the tool by calculating the densities of TEs within Piwi-interacting RNA (piRNA) clusters for both mouse and rat genomes.

We designed a tool called RepCalc that takes as input the annotation information for TEs, which is usually available through a variety of databases and specialized annotation tools. Our tool then generates detailed quantitative report regarding the TE densities within specific ROIs as well as across the whole genome. It is important to note here that this tool is not meant to be used as a genome annotator for the repeats but as a tool that simplifies analysis and comparison of annotated TE information and provides a quick summary of those results.

2. Implementation

RepCalc was developed using Python and requires a separate installation of Tkinter to run the tool in graphical mode in Linux. Tkinter is a standard Python interface to the Tk graphical user interface toolkit [22]. However, the program can run on a Linux terminal without the support of a visual interface if the Tkinter package is not installed. It also can run directly on both Windows and Mac OS X since Tkinter is included during the installation of Python 3.4 or later versions.

3. Results And Discussion

RepCalc has two modes: graphical mode and command line mode. To run RepCalc from the command line, the user must specify a number of arguments. Some of these arguments are mandatory while others are optional. The general format for the order of arguments in the command line mode is shown below. The square brackets indicate optional arguments. However, the optional arguments can sometimes become required, depending upon which arguments are used and the requirements set by the user. Below are the command line arguments needed to run RepCalc.

./repcalc -options [length] TE file [CO] ROI file [CO] output file [configuration]

- TE file: specifies transposable element filename
- CO: column order
- ROI file: specifies region of interest file

RepCalc can be viewed as a set of three distinctive tools that provide the user with a different set of information regarding the quantitative information for the densities of the TEs based upon the user's needs. Each tool gives the user a different set of useful information with reference to the task being performed. The user may choose one of three different analysis options. The first analysis option computes the densities of the TEs within the whole genome (WG). The second analysis option finds the total sum of all the densities but within specific ROI for each family and subfamily of TEs (ROI). The third analysis option calculates the TE densities within individual sub-regions for certain ROI/s for each family and subfamily of TEs (MXROI). Those analysis options correspond to options (a), (b) and (c), respectively, in the command line version of the RepCalc tool. Table 1 below shows the different command line options for Repcalc.

Table 1. Various options for the command line version of RepCalc.

Description of the option	Command line option
Calculates the densities of TEs within the whole genome. This option runs the WG-tool.	A
Calculates the total sum of densities within certain ROIs for each family and	B

subfamily of transposable elements. This option runs the ROI-tool.	
Calculates the transposable element densities within individual ROIs for each family and subfamily of transposable elements. This option runs the MXROI-tool.	C
The user must specify the order of the columns for the TEs file.	A
The user must specify the order of the columns for the RIO file.	B
Transpose the output file (this option only works with the c option)	T

In graphical mode, the analysis options are decided by choosing one of three radio buttons. It is important to note here that the user is required to choose only one of the three analysis options for either mode.

In the command line mode, the options argument defines how the arguments following it shall be used. The option letters are (a), (b) and (c), which represents the different options to calculate TE densities discussed earlier.We also have the (A) and the (B) options which tells the program that the column order for the TE file and the ROI file must be explicitly provided by the user after the name of the file. The TE and ROI files contain information regarding their locations (where they start and end, and the chromosome in which they reside) with a unique ID assigned to each location. The option (A) is for the TE file and option (B) is for the ROI file. If the user does not provide (A) or (B) in the options, then the order of the column for each one of the data elements in the files must match the default column order. The default column order for the data elements is (ID, chromosome, start, and end) which corresponds to column 1, 2, 3 and 4 respectively. TE Start and end within the TE or the ROI file are the location information of the TEs and ROIs. RepCalc can accept a various number of file types input provided that the user specifies the correct column order for each data element and that the data column is whitespace delimited.

It is important that the ID column in the TE files match a specific family/subfamily structure, which is identical to the format given by RepeatMasker. In the ROI file, the ID can represent any unique identifier for the regions of interest. These regions of interest may be genes, piRNAs, miRNAs or any number of regions within the genome. The option (t) in the command line mode, which transposes the data, is only used with the (c) option to transpose the matrix output, to simplify the readability of the output file. The length argument is mandatory when using the (a) and (b) options, which are the length of the whole genome and the total length of the regions of interest, respectively. When using the (c) option, the length should not be provided. The TE file argument is mandatory for all the options while the ROI file argument is mandatory for options (b) and (c).

The final argument in the command line mode is an optional configuration file which is created by the user. The rules to create the configuration file are specified in the readme and example.conf files in the documentation of the program. This option enables the user to modify how the family/subfamily structure of TEs are displayed in the output file. It also enables the user to control which TE families/subfamilies should be included when calculating the total interspersed repeats. In addition, it allows the user to specify whether certain families or subfamilies of TEs are to be included under a different name or under another family of transposable elements. For example, one would use this option if the RC (Rolling-Circle) TE family has a subfamily (i.e. Helitron) and the user does not wish for that subfamily to be listed in the output file as an individual subfamily entry under the main RC family and would prefer it to be listed and counted as part of the main RC family. The user can specify this change in the configuration file by typing RC/Helitron = RC. This will ensure that for each time a Helitron subfamily is encountered under the RC family, it will be reported and listed as part of the RC family without listing the subfamily in the output.

In the graphical mode of the program, based upon the user's choice of the analysis type, certain data fields in the interface will be enabled or disabled. The user must still provide the column order of the data elements for the files uploaded if they do not match the default column order for the data elements discussed earlier.

Below is an example of the input requirements that shows how each tool can be used from within the command line mode:

1. The WG-tool can be used by choosing the (a) option in the command line mode. The length of the genome must be specified. The TE annotation file containing the locations of the TEs within the genome follows. This file can often be downloaded from available databases or can be created by the user using specialized tools. A restriction is that the file must have the order of the columns matching the default column order or the user must explicitly specify the order for each of the columns required to perform the analysis. A name for the output file is specified next. The tool then calculates the density of the transposable elements within the whole genome for that species and produces a table file similar to the one generated by RepeatMasker with the number, length, and percentage for each class and subclass of TEs. The main advantage of using this tool is efficiency and speed since it is possible to generate the table file for the whole genome of an organism in a short time, less than two

minutes in most cases when the TE annotation information is already available.

2. The ROI-tool can be used by choosing (b) in the command line mode. The ROIs can be genes, microRNAs (miRNAs), small interfering RNAs (siRNAs), Piwi interacting RNA (piRNA) or piRNA clusters, or any set of regions that the researcher wants to analyze. The user must supply the TE file following the same rules explained previously for the WG-tool. The user must then specify a file that contains the information regarding the location of the ROI within the genome. This file can be downloaded from online databases, or created by the user. However, the file must follow the default column order for the file or explicitly specify the order of the columns when creating the file. The third argument is the total length of the ROIs. The user then specifies a name for the output filename. The total of all the quantitative information for the densities of the TEs within all the regions is summed, and an output file with the number, length, and percentage for each family and subfamily of transposable elements is produced. Table 2 below shows an example of a sample of the output table file. The running time for this program will vary depending upon the type of data that the user needs to process, whether the ROIs are piRNAs, miRNAs, genes or any other type of data, and the number and size of regions that need to be analyzed.

Table 2. A sample of the table file which shows the output generated using the ROI-sub-tool

Classes and Subclasses of (TEs)	**number of elements**	**length occupied**	**percentage**
SINE:	629772	90756490 bp	8.225%
MIR	30228	3608577 bp	0.327%
Deu	334	35149 bp	0.003%
B4	150860	24400087 bp	2.211%
B2	174238	29468031 bp	2.671%
Alu	256697	31969398 bp	2.897%
ID	17343	1266177 bp	0.115%
No subclass	72	9071 bp	0.001%
LINE:	265284	158098186 bp	14.328%
RTE-BovB	45	3783 bp	< 0.0005%
CR1	2492	361438 bp	0.033%
Penelope	12	2093 bp	< 0.0005%
RTE-X	231	41615 bp	0.004%
L2	15568	2581223 bp	0.234%

3. The MXROI tool can be used by choosing (c) in the command line mode. The TE file, the ROI file, and the output file are then specified. Here, the length of the regions is not required since each individual sub-region is included by default in our calculation. The user may also transpose the data using the (t) in the options argument, and this transpose option is enabled in the graphical mode. This tool gives the user specific information about the quantitative information for the densities of the TEs for each family and subfamily in the specified region/s and writes this information into a file in a matrix form. This tool requires more time than the previous two tools since it provides detailed output regarding each individual region. The running time for this program varies depending on the type of data being processed, the number of regions, and the size of the genome. Figure 1 below shows the various features of RepCalc and how the user can choose different analysis options.

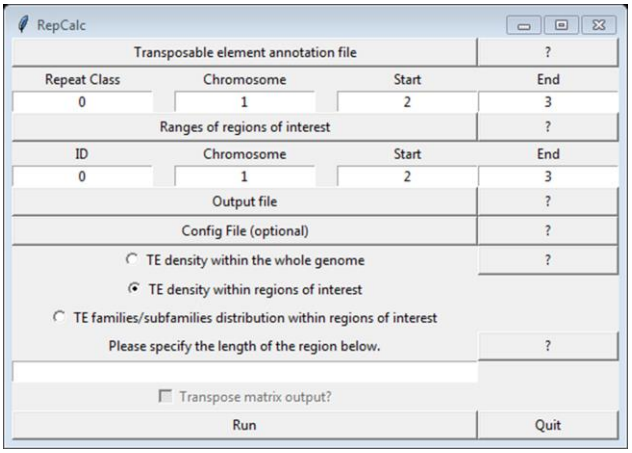


Figure 1: Different features within RepCalc

4. Application On Real Data

To test our tool we chose the region of interest to be piRNA clusters. The locations of piRNA clusters vary between different species as well as within different databases, depending upon the mechanism used to identify those clusters. We chose to analyze two main databases containing piRNA data: piRNA Bank [23] and Johannes Gutenberg University of Mainz piNRA database (JGU database) [24]. The number of clusters within the same organisms varies between those two databases. In piRNA bank, the number of mouse piRNA clusters is 2710 while in the JGU database it is 171, also the number of piRNA clusters of rat is 189 in piRNA bank and 168 in JGU database. The difference in piRNA clusters in both databases could be attributed to the different methodologies used by each database to predict piRNA clusters.

From each database, we extracted the information necessary to locate each piRNA cluster. The information usually includes the cluster ID, the chromosome where it

is located, and the starting and ending position for the cluster. We provided each tool with the necessary files. For the WG tool, this is the TE locations found in the annotation file and total genome length. For the ROI tool, we provided the piRNA cluster locations, the TE annotation information and total length for the ROIs. For the MXROI tool, we used the same information that was used for the ROI tool with the exception of the information regarding the length of the ROI.

We ran the tools on the piRNA data and generated table files representing the densities for the TEs within the whole genome for both mouse and rat using the WG tool. Using the ROI tool we obtained the densities for the TEs within piRNA clusters. We also created a matrix representing the densities for the TEs for each family and sub-family of transposable elements within each distinct piRNA cluster using the MXROI tool. To verify the accuracy of our calculation we compared the densities for the TEs within the genome calculated using our tool to the densities calculated using Repeat Masker. Figures 2 and Figure 3 below show the different densities for TEs across different regions within the mouse and rat genome.

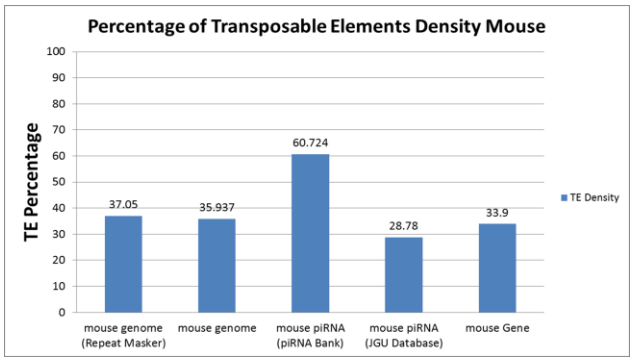


Figure 2: Different Densities of TE in Mouse

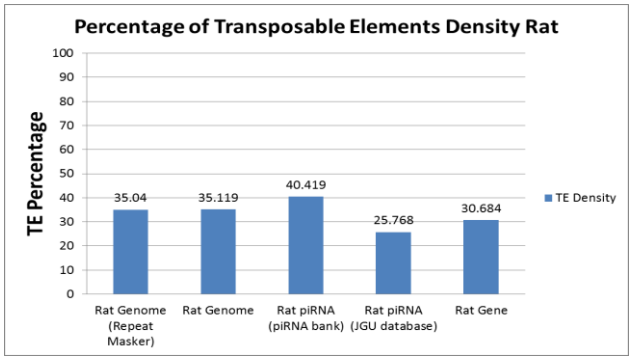


Figure 3: Different Densities of TE in Rat

5. Conclusions

RepCalc is a tool designed to be used for generating detailed quantitative information about the distribution of certain classes of TE families and their subfamilies at the genome level, within specific ROIs and across individual sub-regions, utilizing TE annotation information available throughout the various databases and annotation tools designed for this purpose. The tool is comprised of three tools, each one of them provides the user with a different level of quantitative detail regarding the TE densities across different segments of the genome.

To demonstrate how the tool works, we calculated the densities of TE families and sub-families within piRNA clusters and compared them to the TE densities within the genes and across the genomes of both mouse and rat utilizing two different piRNA databases. We found that the piRNA clusters that were extracted from piRNA bank exhibit a higher TE density than the JGU database for both the mouse and rat genes as well as across the whole genome.

Availability And Requirements

Project name: RepCalc
Project home page: https://github.com/eenblam/repcalc
Operating system: Windows, Linux, Mac OS X
Programming languages: Python 3.4
Other requirements: Python 3.4, Tkinter

Competing Interests

The authors declare that they have no competing interest.

Authors' Contributions

TA, AP and FH developed the concept. TA and BE designed the software. BE implemented the software. TA and BE tested and evaluated the software. TA, AP and BE wrote the manuscript. All the authors read and approved the final manuscript.

Acknowledgements

This work was supported by the National Science Foundation under award EPS-0903787. In addition we would like also to thank Dr. David Ray for his helpful suggestions throughout the work on this project.

References

[1] Lander ES, Linton LM, Birren B, Nusbaum C, Zody MC, Baldwin J, et al. Initial sequencing and analysis of the human genome. *Nature*, 409:860–921, 2001.

[2] Pray L. Transposons: The jumping genes. *Nat. Educ*,1:204, 2008.
[3] Fedoroff N V. Transposable Elements, Epigenetics , and Genome Evolution. *Science*, 338:758–767, 2012.
[4] Malone CD, Hannon GJ. Molecular evolution of piRNA and transposon control pathways in Drosophila. Cold Spring Harb Symp *Quant Biol*, 74:225–234, 2009.
[5] Slotkin RK, Martienssen R. Transposable elements and the epigenetic regulation of the genome. *Nat Rev Genet*, 8:272–285, 2007.
[6] McClintock B. The Origin and Behavior of Mutable Loci in Maize. *Proc. Natl. Acad. Sci. U. S. A.*, 36:344–355, 1950.
[7] Bergman CM, Quesneville H. Discovering and detecting transposable elements in genome sequences. *Brief. Bioinform.*,8:382 – 392, 2007.
[8] Lerat E. Identifying repeats and transposable elements in sequenced genomes: how to find your way through the dense forest of programs. *Heredity*,104:520 – 533. 2010.
[9] Smit, AFA, Hubley, R & Green P. RepeatMasker. http://www.repeatmasker.org/, March 2017.
[10] Jurka J, Klonowski P, Dagman V, Pelton P. CENSOR--a program for identification and elimination of repetitive elements from DNA sequences. *Comput. Chem*, 20:119 –121.1996.
[11] Volfovsky N, Haas BJ, Salzberg SL. A clustering method for repeat analysis in DNA sequences. *Genome Biol*, 2(8):1-11, 2001.
[12] Achaz G, Boyer F, Rocha EPC, Viari A, Coissac E. Repseek, a tool to retrieve approximate repeats from large DNA sequences. *Bioinformatics*, 23:119–121, 2007.
[13] Estill JC, Bennetzen JL. The DAWGPAWS pipeline for the annotation of genes and transposable elements in plant genomes. *Plant Methods*,5(1):1-11, 2009.
[14] Price AL, Jones NC, Pevzner PA. De novo identification of repeat families in large genomes. *Bioinformatics*, 21:i351–i358, 2005.
[15] Jurka J, Kapitonov V V., Pavlicek A, Klonowski P, Kohany O, Walichiewicz J. Repbase Update, a database of eukaryotic repetitive elements. *Cytogenet. Genome Res*,110:462–467 ,2005.
[16] Carène Rizzon, Gabriel Marais, Manolo Gouy, Christian Biémont. Recombination Rate and the Distribution of Transposable Elements in the Drosophila melanogaster Genome. *Genome Res*, 12: 400-407, 2002.
[17] Cridland JM, Macdonald SJ, Long AD, Thornton KR. Abundance and distribution of transposable elements in two Drosophila QTL mapping resources. *Mol Biol Evol*, 30(10):2311-2327, 2013.
[18] Bartolomé C, Maside X, Charlesworth B. On the abundance and distribution of transposable elements in the genome of Drosophila melanogaster.*Mol Biol Evol*. 19(6):926-937, 2002.
[19] Jae-Sung Rhee, Beom-Soon Choi, Jaebum Kim, Bo-Mi Kim, Young-Mi Lee, Il-Chan Kim, Akira Kanamori, Ik-Young Choi, Manfred Schartl & Jae-Seong Lee. Diversity, distribution, and significance of transposable elements in the genome of the only selfing hermaphroditic vertebrate Kryptolebias marmoratus. *Scientific Reports*. 7, Article number:40121, 2017.
[20] Leonardo Galindo González and Michael K Deyholos. Identification, characterization and distribution of transposable elements in the flax (Linum usitatissimum L.) genome. *BMC Genomics*. 13:644, 2012.
[21] Xulio Maside, Carolina Bartolome, Stavroula Assimacopoulos, Brian Charlesworth. Rates of movement and distribution of transposable elements in Drosophila melanogaster: in situ hybridization vs Southern blotting data. *Genetics Research*. 78(2):121-136, 2001.
[22] 24.1. Tkinter — Python interface to Tcl/Tk — Python v2.7.8 documentation. https://docs.python.org/2/library/tkinter.html, August 2014.
[23] Lakshmi SS, Agrawal S. piRNABank: a web resource on classified and clustered Piwi-interacting RNAs. *Nucleic Acids Res*, 36:D173–D177, 2008.
[24] Rosenkranz D, Zischler H. proTRAC - a software for probabilistic piRNA cluster detection, visualization and analysis. *BMC Bioinformatics*,13:1-10, 2012.
[25] Laurent Duret, Gabriel Marais , Christian Biémont. Transposons but Not Retrotransposons Are Located Preferentially in Regions of High Recombination Rate in Caenorhabditis elegans. *GENETICS*, 156(4): 1661-1669, 2000.
[26] Cas Simons, Michael Pheasant, Igor V. Makunin, John S. Mattick. Transposon-free regions in mammalian genomes. *Genome Res*, 16(2): 164–172, 2006.

Practical Space-efficient Linear Time Construction of FM-index for Large Genomes

Elena Y. Harris
California State University, Chico
Chico, California, 95929, USA
eyharris@csuchico.edu

Abstract

The Burrows-Wheeler Transform (BWT) and Full-text index in Minute space (FM-index) are indispensable data structures that are used in the next generation sequencing data analysis to efficiently map reads to a reference genome. Recently developed algorithms SA-IS and BWT-IS allowed construction of a Suffix Array and Burrow-Wheeler Transform, respectively, for mammalian-size genomes in less than an hour. In practice, BWT-IS algorithm outperforms SA-IS in terms of RAM usage. Building an FM-index from a BWT requires LF-mapping that is a relatively time-consuming step. Here, we present a space-efficient linear time algorithm called BWT-ISFM that builds an FM-index concurrently with a construction of BWT. Our algorithm supports a genome size of up to 8Gb (giga-basepairs) in length while a publicly available BWT-IS has a limit on a genome size of 4Gb. Moreover, in practice, our algorithm requires only 2.3*n* bytes of RAM for a genome size of *n* compared to 4*n* bytes of RAM used by SA-IS algorithms.

keywords: next generation sequencing; BWT; FM-index

1 Introduction

The Burrows-Wheeler Transform, BWT, introduced by Burrows and Wheeler [1] together with a Full-text index in Minute space, FM-index, proposed by Ferragina and Manzini [3], are used to align sequenced reads generated by the next generation sequencing instruments to a reference genome. Suffix Array, SA, a data structure introduced by Manber and Myers [9] is a concise representation of sorted suffixes of a given string T. Given a string T, SA is an array that stores positions of the suffixes of T sorted in lexicographic order. Given a SA, BWT can be constructed in linear time by scanning the SA and retrieving characters of T at the previous positions of the stored in the SA positions. Recent advances in linear time construction of a SA (see the survey by Dhaliwal et al. [2]) allow building a SA for large mammalian genomes such as human genome in tens of minutes using practically affordable space of about 15GB. It has been shown in Harris et al. [4] that building the BWT on two strands of a genome, positive strand and negative strand, speeds up read alignment two times. In this case and, in general, for larger genomes, space of at least 4*n* bytes (hereafter, *n* is the length of a genome) required by the most time-efficient SA algorithms, becomes not so practical. For example, building a SA over two strands of a human genome of 6Gb would require at least 24GB.

Only few aligners use a SA to map sequenced reads to a reference genome, e.g. Hoffmann et al. [5]. Most aligners rely on using BWT together with a FM-index. Recently, a linear time algorithm BWT-IS for building BWT directly was developed by Okanohara and Sadakane [11]. It extends on the ideas of the recursive linear time SA-IS algorithm by Nong et al. [10] that uses induced sorting to build a suffix array. In practice, BWT-IS requires only up to 2.2*n* bytes of memory, which is a significant advantage over 4*n* required for building a SA. Building BWT directly (without building a SA) for a human genome size is also time efficient – it takes less than a half of an hour on a human genome of size 3Gb. To build an FM-index from BWT, first, a succinct data structure is built in linear time that stores for each position *i* of BWT the total number of occurrences of each character of the given alphabet in BWT from the beginning to *i*. This structure is used in the next step, LF-mapping, to retrieve and explicitly store genomic positions equidistant from each other at a fixed range, called *step*. In practice, O(*n*)-time LF-mapping is relatively time consuming (compared to the building of the BWT). Some algorithms (e.g. by Kärkkäinen [6]) build BWT and SA in consecutive blocks, so the genomic positions for each block are available for output on the fly. The downside of this approach is $\Omega(n\log n)$ time requirement for building the BWT and FM-index. The linear time algorithm BWT-IS does not store genomic positions while constructing a BWT. In order to construct an FM-index, we must first build BWT and then use LF-mapping to construct an FM-index, which requires additional time. Moreover, the existing implementation of BWT-IS provided by the authors and publicly available implementations of SA-IS have a limitation on the genome size of less than 4Gb.

Here, we propose a practical, space-efficient, linear time algorithm, BWT-ISFM, that calculates the FM-index while building the BWT directly from a given genome. For two-strands of human genome of 6Gb, it requires only 2.3*n* bytes of memory and builds the BWT and FM-index in less than an hour. Our algorithm supports genome size of up to 2^{33} characters long, 8Gb. In addition, it introduces a new algorithmic approach to optimize one step of SA-IS algorithm that may be used for a pool of ideas on optimization of BWT and SA constructions for small,

978-1-943436-11-8 / copyright ISCA, BICOB 2018

March 19-21, 2018, Las Vegas, Nevada, USA

constant alphabets. Thus, our algorithm offers a memory advantage over SA-IS algorithms, extends genome size limitation, and constructs an FM-index on the fly while building the BWT in practical time.

2 Preliminaries and Related Work

Let T be a string of length n over the given ordered alphabet Σ of size σ. Let $ be the sentinel character that is the smallest in Σ and occurs in T only once at the end of the string. Let T[i] be a character of T, for $0 \le i \le n$-1, and T[$i…j$] be a substring of T of consecutive characters starting with character T[i] and ending with T[j]. A suffix of T, denoted by T_i, is a substring T[$i…n$-1], i.e. T_i ends with $. A suffix array SA for a given string T is an array of size n such that SA[i] stores the starting position p of suffix T_p for $0 \le i \le n$-1 and $T_{SA[0]} < T_{SA[1]} < … < T_{SA[n-1]}$. In other words, SA holds the starting positions of all suffixes of T sorted in lexicographic order. Given a string T of length n over alphabet Σ of size σ, a suffix array can be built in linear time. Some of the recently developed algorithms for SA construction that use recursion and induce sorting of suffixes are by Ko and Aluru [7] and by Nong et al. [10]. Hereafter, we will refer to the algorithm by Nong et al. [10] called SA-IS in our detailed discussion.

SA-IS uses a concept of LMS substrings to induce-sort suffixes of a given string T. To understand an LMS substring, we need to categorize characters of T by L-type and S-type (stands for Large and Small). The sentinel character $ is of S-type. For $0 \le i \le n$-2, character T[i] is of S-type if it is lexicographically smaller than the next character T[i+1], and T[i] is of L-type if it is greater than T[i+1]. If T[i] and [i+1] are equal, then T[i] has the same type as T[i+1]. Enumeration of characters of T by S- or L-type can be done in linear time by scanning T in Right-to-Left fashion. An LMS character stands for the left most S-type character and it is character T[i] of S-type that has previous character T[i-1] of L-type. Character T[0] is considered to have sentinel as the previous character, so T[0] cannot be an LMS character. Further, suffixes of T are named L-type, S-type and LMS-type after their first character's type, e.g. if T[i] is of L-type, then T_i is an L-type suffix. An LMS substring of T is a substring T[i…j] such that T[i] and T[j] are LMS characters and no other character between indices i and j are LMS characters. The sentinel character is the only LMS substring of length one.

When suffixes of T are arranged in lexicographic order, suffixes starting with the same character c occur in consecutive range, and their positions are stored in consecutive entries of SA. We will refer to the consecutive range SA[i…j] that stores starting positions of all suffixes of T starting with character c as a c-bucket. Sorted in lexicographic order suffixes in the same c-bucket have the following order: L-type suffixes precede S-type suffixes in the c-bucket (please refer to the original paper for proof of this and other statements regarding SA-IS algorithm).

Furthermore, SA-IS algorithm uses a notion of fronts and ends of c-buckets. If a SA[i…j] is a c-bucket, initially, index i is the head of c-bucket pointing to the front of the bucket and index j is the tail of c-bucket pointing to the end of the bucket; as entries of SA fill in, the heads and the tails of the buckets are incremented and decremented respectively.

Figure 1 shows the outline of SA-IS algorithm and Figure 2 demonstrates the execution of the steps of SA-IS for a string T='DABRACADABRACABRAB$'. Given a string T, SA-IS calculates the type array t that stores S- and L-type for characters of T. Using t, it identifies the starting positions of LMS substrings and places them into ends of the corresponding c-buckets. This is done in linear time by scanning t Right-to-Left (see Figure 2, A and B). Next, *InduceSort*(T, SA, t) procedure consists of two steps: (1) Left-to-Right scanning of SA with head pointers initialized to the fronts of the corresponding c-buckets, and (2) Right-to-Left scanning of SA with tail pointers initialized to ends of the corresponding c-buckets. During Left-to-Right scanning of SA, for each position p at SA[i], it checks whether the character T[p – 1] at the previous position p – 1 is of L-type, and if so, it places position (p – 1) into the current front of the c-bucket, where c is T[p – 1], and increments the head of c-bucket (see Figure 2 C).

During Right-to-Left scanning of SA, for position p at SA[i], it checks whether the character T[p – 1] at the previous position p – 1 is of S-type, and if so, it places position (p – 1) into the current end of the c-bucket, where c is T[p – 1], and decrements the tail of c-bucket. By the end of this step, LMS substrings are correctly sorted in lexicographic order relative to each other.

Figure 2 D shows SA after this step and shows LMS substrings sorted in lexicographic order relative to each other.

ALGORITHM 1: SA by induced sorting
0: **Input**: string T of length n over alphabet Σ of size σ
Output: Suffix Array for T
1: Check for termination condition: if n is equal to σ, calculate SA directly
2: Calculate S/L-type array t
3: Place the starting positions of LMS substrings into the ends of c-buckets of SA
4: *InduceSort*(T, SA, t)
5: Assign names to LMS substrings
6: Build the shortened string T_1
7: Recursively calculate SA_1 for T_1
8: Induce positions of LMS substrings of T from positions of suffixes T_1 stored at SA_1
9: Place the starting positions of LMS substrings in the sorted order in their corresponding c-buckets of SA
10: *InduceSort*(T, SA, t)
11: **return**: SA

Figure 1: SA-IS outline

The next step of SA-IS is to assign a new integer-name to each LMS substring. The sentinel LMS is assigned name 0. The naming of the rest of LMS substrings is done by Left-to-Right scanning of SA, and comparing two consecutive LMS substrings: if two substrings are the same, then they are assigned the same name, otherwise, the current LMS is assigned the next integer-name than the previous LMS.

Finally, to build a shortened string T_1 that consists of integer-names of LMS substrings of T, the integer-names must be placed in the same order as their corresponding LMS substrings occur in T. To clarify, if the *i*-th LMS substring of T has been assigned integer-name d_1 and the *k*-th LMS substring of T has been assigned integer-name d_2, then character $T_1[i] = d_1$ and $T_1[k] = d_2$. Figure 1 E shows the resulting T_1 for the given T in our example.

To make sure this step is done in linear time, the original algorithm [10] proposed to keep a bit array of length *n* with 1s denoting the starting positions of LMS substrings. In addition, a succinct data structure supporting *Rank*(*i*) operation must be prebuilt in linear time, where given the starting position *i* of an LMS substring in T, *Rank*(*i*) in constant time returns the rank of the LMS substring in T (the order of the LMS substring in T from left to right). Thus, once the LMS substring T[i…j] is assigned name d, we set $T_1[\ Rank(i)\ ] = d$.

The recursive call to SA-IS on the shortened string T_1 returns the suffix array SA_1 for T_1. The next step of the algorithm is to induce the positions of LMS substrings of T from the suffix positions of T_1 stored in SA_1. Since each character $T_1[j]$ corresponds to the *j*-th LMS substring of T, we can convert positions of suffixes of T_1 to starting positions of the corresponding LMS of T as follows. Let $j = SA_1[i]$ be the position of the suffix of T_1, starting with $T_1[j]$, which corresponds to the *j*-th LMS in T. Then we can use a prebuilt in linear time succinct data structure that supports operation *Select*(*j*) that returns the position of the *j*-th LMS in T in constant time, given *j*. After inducing the starting positions of the LMS substrings in T from suffix positions of T_1, we have positions of LMS suffixes of T sorted in lexicographic order and place them into SA at the ends of the corresponding *c*-buckets.

The last step *InduceSort* is used again to induce SA from the LMS positions. Each step of SA-IS takes linear time. Since two LMS characters cannot be consecutive characters by definition, then the length of T_1 is at most *n*/2, half of the length of T, and, hence, the recursive algorithm SA-IS takes linear time.

A

index *i*	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
T	D	A	B	R	A	C	A	D	A	B	R	A	C	A	B	R	A	B	$
t	L	S	S	L	S	L	S	L	S	S	L	S	L	S	S	L	S	L	S
LMS		*			*		*		*			*		*			*		*

B

c-bucket	$	A							B				C		D		R		
index *i*	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
SA	18	1	4	6	8	11	13	16											

C

c-bucket	$	A							B				C		D		R		
index *i*	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
SA	18	1	4	6	8	11	13	16	**17**				**5**	**12**	**0**	**7**	**3**	**10**	**15**

D

c-bucket	$	A							B				C		D		R		
index *i*	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
SA	**18**	**16**	**1**	**8**	**13**	**4**	**11**	**6**	17	**2**	**9**	**14**	5	12	0	7	3	10	15
	$	A	A	A	A	A	A	A											
		B	B	B	B	C	C	D											
		$	R	R	R	A	A	A											
			A	A	A														

E

T_1

rank of LMS *i*	1	2	3	4	5	6	7	8
integer-name	2	3	4	2	3	2	1	0

Figure 2: Execution of Algorithm 1 applied to the given string T='DABRACADABRACABRAB$'. (A) Type array *t* is shown for T and the starting positions of LMS substrings are marked with * character. (B) Suffix array SA is shown after the starting positions of LMS substrings have been placed at the end of the *c*-buckets, where *c* is the starting character of an LMS. (C) SA is shown after Left-to-Right scanning of SA and after the order of L-type suffixes has been induced from LMS and L-type suffixes. The induced L-type suffix positions are shown in bold. (D) SA is shown after Right-to-Left scanning and after the order of S-type suffixes has been induced from L-type and S-type suffixes. The induced S-type suffix positions are shown in bold. (E) The shortened string T_1 is shown: each integer-character $T_1[i]$ corresponds to the *i*-th LMS substring of T, whose names have been assigned according to their lexicographic order in SA.

BWT-IS simulates *InduceSort* procedure with the help of four queues: LMS, L, S and LS (each of the four queues for each character in the alphabet). Instead of keeping a suffix array of size *n* that holds positions of all suffixes, BWT-IS keeps LMS substrings of T directly and uses circular shift of characters in LMS so that the front character directs the next step of induce-sorting. Initially all LMS substrings are reversed and placed into LMS queues (by their last character), e.g. the reverse of an LMS T[i…j] is stored in $LMS_{T[j]}$ queue for T[j] character.

Left-to-Right scanning of *InduceSort* procedure processes *c*-buckets in increasing order of characters *c*. BWT-IS simulates this step by considering characters of the alphabet in increasing order, and for each character *c*, first, it processes L_c queue and then LMS_c queue (just as SA-IS algorithm processes L-type suffixes and then LMS suffixes of a *c*-bucket). While L_c queue is not empty, a current LMS is popped at the front of the queue, and if the front character *b* of the current LMS substring is greater than or equal to *c,* then *b* is of L-type, so the LMS substring is pushed to the back of L_b queue (in this case, *b* is shifted to the back of the LMS substring). Otherwise, the LMS substring is pushed to the back of LS_c queue. This simulates processing of L-type suffixes of a *c*-bucket. Next, while LMS_c queue is not empty, a current LMS substring is popped from the front of the queue, and the front character *b* of the current LMS substring is placed to the back of the LMS substring. The character *b* is of L-type, so the LMS substring is pushed to the back of the L_b queue. These two steps simulate Left-to-Right scanning of SA and induce sorting of L-type suffixes.

The Right-to-Left scanning of SA of SA-IS algorithm is simulated by processing characters of the alphabet in decreasing order. Prior to this step, LS queues are reversed. Next, for each character *c*, first S_c queue is processed and then LS_c queue is processed. In SA-IS this corresponds to processing of S-type suffixes and then L-type suffixes of a *c*-bucket during Right-to-Left scanning of SA. While S_c queue is not empty, pop a current LMS substring from the front of the queue, move the front character *b* of the LMS substring to the back of the LMS substring, and if *b* is less than or equal to *c* (i.e. *b* is of S-type), then push the LMS substring onto the back of S_b queue. Next, while LS_c is not empty, pop a current LMS substring from the front of the queue, move the front character *b* to the back of the substring, and push LMS substring onto the back of S_b queue. During these movements involving the four types of queues, BWTs of L-type suffixes and of S-type suffixes are built separately for each character *c*, and at the end of the algorithm, BWT for T is constructed from these shorter BWT substrings.

Hereafter, we will refer to BWT-IS as it is implemented by the authors of the original paper. Given a string T, (1) BWT-IS sorts LMS substrings using *quick sort* and builds a shortened string T_1, then (2) calls SA-IS algorithm to recursively calculate SA_1 for T_1; (3) deduces positions of the LMS substrings of T from positions of suffixes of T_1 in SA_1; and finally, (4) simulates *InduceSort* using the four types of queues to build the BWT for T. BWT-IS saves space by avoiding storing SA for T.

In the presented here algorithm, we use BWT-IS as the basis for our algorithm BWT-ISFM. The major difference between the proposed algorithm and BWT-IS is that we keep the starting positions of LMS suffixes in the queues instead of LMS substrings. This allows accessing induced-sorted positions of suffixes directly on the fly while constructing the BWT, which allows building an FM-index on the fly. In addition, our implementation of sorting distinct LMS substrings and building a shortened T_1 of the given string T differs from the BWT-IS's implementation. The rest of the paper is organized as follows. In section 3, we describe our algorithm and analyze its time requirements. In section 4, we convey a benchmarking that demonstrates the performance of our algorithm in terms of time and RAM, and compares it with the performance of BWT-IS and existing FM-index building tools: the most popular tool Bowtie-2 by Langmead et al. [8] and another tool called BRAT-nova by Harris et al. [4], an aligner for bisulfite-treated reads used to identify methylation within a DNA sequence.

3 Construction of BWT and FM-Index

3.1 Implementation of Induce Sorting and Calculation of Explicitly Stored Positions

FM-index constructed by our algorithm BWT-ISFM consists of (1) a succinct data structure *Character Occurrences* that for each character *c* in Σ allows calculating of the total number of occurrences of *c* in BWT[0…*i*] in constant time, for $0 \le i \le n-1$; (2) a succinct data structure *Positions Occurrences* that consists of a bit array called *bwtMarked* (with bit 1 at index *i* indicating that the suffix position corresponding to SA[*i*] is explicitly stored) together with a succinct structure that calculates *Rank*(*i*) in constant time; this structure is used to retrieve an explicitly stored genomic position; (3) an array called *Positions* with explicitly stored suffix positions. In addition to this classical FM-index, our program constructs a bit array called *posBit* with 1 at index *i* indicating that stored position corresponding to SA[*i*] is greater than maximum value of an unsigned integer, MAXUI = $2^{32} - 1$.

The array *Positions* stores unsigned integers (requiring 4B per integer). To retrieve the correct position, algorithm uses 4B stored at *Positions* and one bit stored at *posBit*: if a bit at *posBit* is 1, then to the value stored at *Positions*, we need to add MAXUI.

Our algorithm follows the outline of BWT-IS. Our algorithm uses four types of deques: LMS, L, S and LS (similarly to the queues used by BWT-IS described above). Each deque supports four operations: push front and push back (inserts a suffix position at front and back respectively), and pop front and pop back (removes a suffix position from the front and back of a deque respectively). At any time of the algorithm, there are at most X positions stored in all deques, where X is the total number of LMS substrings of the original string T.

The procedure *InduceSort* of BWT-ISFM is shown in Figure 3. As in BWT-IS, *InduceSort* is used only once. We keep BWT array of length $2n$ bits (2 bits per character). Initially, the starting positions of sorted LMS substrings of T are pushed back onto the corresponding LMS_c deques for each character c, the starting character of an LMS substring; and the head pointers for each character c in Σ are set to the fronts of corresponding c-buckets. The moves of LMS positions between the deques exactly simulate induce sorting using an SA in SA-IS algorithm. First, characters of alphabet are processed in increasing order in the first ***for*** loop (simulating processing of c-buckets in Left-to-Right order), and then characters are processed in decreasing order in the second ***for*** loop (simulating processing of c-buckets in Right-to-Left order). In the first *for* loop, L_c deque is processed before LMS_c deque. In each of these deques, a current front position p is popped from a deque, and if the character T[p-1] is greater or equal to T[p], then T[p-1] is of L-type, and position $(p-1)$ is pushed back onto $L_{T[p-1]}$ deque. The corresponding BWT character is calculated as T[p-2]. In addition, we check whether the position $(p-1)$ is a position that we explicitly store (mod *step* is equal to 0, where *step* is $log(n)$). If so, then we output to a file the position $p-1$ and the BWT index, $head_{T[p-1]}$. In addition, we set bits of *bwtMarked* (if position $p-1$ is explicitly stored) and *posBit* (if $p-1$ is greater than MAXUI) to 1. At the end, the *head* pointer is incremented. In case when the front of L_c deque induces position $p-1$ that corresponds to S-type suffix, then our algorithm pushes p to the back of $LS_{T[p]}$ deque.

To simulate Right-to-Left scanning of SA of SA-IS algorithm, our algorithm sets *tail* pointers to the back of c-buckets for each character c in Σ. Next, BWT-ISFM processes the second *for* loop, in which for each character c taken in decreasing order, S_c deque is processed first and then LS_c deque is processed. Until a deque is not empty, a current suffix position p from the back of the deque is popped (scanning Right-to-Left), and if the previous position, $p-1$, is of S-type, then the position $p-1$ is pushed to the front of $S_{T[p-1]}$ deque. In addition, BWT character T[p-2] is set at index $tail_{T[p-1]}$ and if needed bits of

ALGORITHM 2: *InduceSort* of BWT-ISFM

```
0: Input: string T and for each character, four
     types of deques: LMS, L, S and LS;
     LMS deques are initialized with positions of LMS
     substrings
   Output: BWT, bwtMarked, posBit, explicitly stored
     positions of suffixes of T
1: for each character c := 0,1,2,…,σ-1, do
2:    for Qc := Lc, LMSc
3:       while Qc is not empty do
4:          p ← Qc.popFront()
4:          if(T[p] ≤ T[p-1]) then
5:             LT[p-1].pushBack( p-1 )
6:             BWT[headT[p-1]] := T[p-2]
7:             if( (p-1) mod step = 0 )
8:                bwtMarked[headT[p-1]] := 1
9:                write: (p-1) and headT[p-1]
10:            if(p – 1 > MAXUI)
11:               posBit[headT[p-1]] := 1
12:            headT[p-1] := headT[p-1] + 1
13:         else if processing Lc queue,
                  and T[p] > T[p-1],
                  then LST[p].pushBack( p )
14:      end of while
15:   end of for
16: end of for
17: for each character c := σ-1, …, 2, 1, 0, do
18:    for Qc := Sc, LSc
19:       while Qc is not empty do
20:          p ← Qc.popBack()
21:          if(T[p-1] ≤ T[p]) then
22:             ST[p-1].pushFront( p-1 )
23:             BWT[tailT[p-1]] := T[p-2]
24:             if( (p-1) mod step = 0 )
25:                bwtMarked[tailT[p-1]] := 1
26:                write: (p-1) and tailT[p-1]
27:             if(p – 1 > MAXUI)
28:                posBit[tailT[p-1]] := 1
29:             tailT[p-1] := tailT[p-1] – 1
30:      end of while
31:   end of for
32: end of for
```

Figure 3: *InduceSort* of BWT-ISFM

bwtMakred and *posBit* at index $tail_{T[p-1]}$ are set to 1, and an explicitly stored position $p-1$ together with index $tail_{T[p-1]}$ are printed to a file. At the end, the *tail* pointer is decremented.

Once *InduceSort* procedure is finished, we need to place explicitly stored SA positions in the correct order, i.e. in increasing order of BWT indices. First, BWT-ISFM pre-builds in linear time the succinct data structure *Positions Occurrences* using *bwtMakred*. Recall that given a BWT index i, this structure calculates $Rank(i)$ in constant time. Then, our algorithm reads in the outputted to the file

indices one at a time (suffix position and the corresponding BWT index). Using the BWT index *i*, it places the corresponding suffix position into the array *Positions* at index *Rank*(*i*). We chose to output positions and their BWT indices into the file to save memory. There are total of $n/log(n)$ explicitly stored positions, and we need $log(n)$ bits to store each position or BWT index. Hence, the total space to store positions and BWT indices would be $2 \cdot log(n) \cdot n/log(n)$, which results in $2n$ bits. In case of keeping these positions and indices in memory, I/O operations would not affect linear time of the algorithm. In practice, I/O operations do not add much to the total time (not more than about 3 minutes in our experiments – results are not shown), but printing out positions and BWT indices allows saving memory for larger genomes and offers extra flexibility for users (our program allows users to select the value for *step*, which regulates the total number of explicitly stored positions).

3.2 Sorting Distinct LMS Substrings, Assigning Names and Building a Shortened String T_1

In SA-IS, the *InduceSort* procedure is used for the first time to sort LMS substrings in lexicographic order relative to each other. One can use *InduceSort* that guarantees theoretical linear time, but in practice, another method to sort all distinct LMS substrings is much faster. For example, in our experiments on a human genome using two strands, *InduceSort* takes about 20 minutes, whereas sorting distinct LMS substrings using a *quick sort* and building a shortened T_1 string takes less than 2 minutes. Here, we will describe our method of building a shortened T_1 using a *quick sort*. We used ideas similar to those of the BWT-IS algorithm, but devised a different implementation for this procedure.

First, we collect the starting positions of distinct LMS substrings into a separate array. LMS are categorized as short substrings and long substrings dependent on their lengths. An LMS substring of length at most 12 characters is considered to be short, and the rest of LMS substrings are long. We would like to clarify that by the term all *distinct* LMS substrings we mean all long LMS substrings and distinct short LMS substrings. To identify the starting positions of distinct short LMS substrings, we keep a hash table of size 4^{12} entries, where 4 is the size of the DNA alphabet {A, C, G, T}. The length 12 for a short LMS substring was chosen to keep a good balance between the space required for the hash table and the total number the long LMS substrings that is at most $n/12$. Each character is represented using 2 bits (A is 00, C is 01, G is 10 and T is 11). We scan T Right-to-Left, and if T[*i*] is the starting character of a short LMS substring *s*, then we convert *s* to its hash index *h* equal to the complement of *s*. We must use the complement of *s* to distinguish between LMS substrings such as AATA and ATA; since A is represented as 0 in binary, both strings AATA and ATA in binary are represented as the same integer, namely, 12 (00001100 and 001100 respectively). By taking the complement of these strings, their binary representations become distinguishable (11110011 and 110011 respectively). For all identical short LMS substrings whose hash index is *h*, we store a single position that is the greatest. For example, if T[i…i+m] and T[j…j+m] correspond to a short LMS substring *s*, and the position *i* is greater than *j*, then we store *i* at index *h* in the hash table and in the array with the distinct LMS positions. If a current LMS is long, we store its starting position in the array with distinct positions.

ALGORITHM 3: Building a shortened T_1

0: **Input**: string T, integer arrays *positionsDistinctLMS* and *namesDistinctLMS*, integer array *hashTable*
Output: shortened string T_1 of size equal to the total number of LMS substrings in T
1: Scan T Right-to-Left:
keep pointer *p* to point to the last position in *positionsDistinctLMS*;
fill in T_1 Right-to-Left using index *j*
2: **If** index *i* is the starting position of LMS substring *s*
3: **If** *i* is equal to *positionsDistinctLMS* [*p*],
4: then $T_1[\ j\ ]$ = *namesDistinctLMS*[*p*];
j--; *p*--;
5: **If** current LMS substring *s* is short,
hashTable[~s] = *namesDistinctLMS*[*p*]
6: **Else if** $i \neq$ *positionsDistinctLMS* [*p*]
7: then $T_1[\ j\ ]$ = *hashTable*[~s]; *j*--;
8: **return**: T_1

Figure 4: Building a shortened T_1

Once the positions of distinct LMS substrings are collected, we sort the distinct LMS substrings using a *quick sort*. Dependent on implementation, theoretically, it takes $\Omega(n \log n)$ time, but in practice, it is much faster than *InduceSort* that takes theoretical linear time (*e.g.* it takes on average 9 seconds to sort all distinct LMS substrings of two strands of human genome). Next, we assign names to the distinct LMS substrings by scanning the sorted array and comparing two consecutive LMS substrings. If the current LMS substring is the same as previous one, it is assigned the same integer-name, otherwise, it is assigned the next integer-name, starting with integer-name of 0 for the sentinel character. The integer-names are stored in another array such that the corresponding starting position of an LMS and its integer-name are stored at the same index of the corresponding arrays.

Finally, to build a shortened string T_1 whose characters are integer-names of LMS substrings of T, we use another *quick sort* that sorts the integer-names of all distinct LMS substrings according to the increasing order of their corresponding starting positions in T. To make it clear, we use *quick sort* to sort pairs <position, integer-name> in increasing order of positions. This places integer-names

into appropriate slots within T_1. Figure 4 shows the procedure that builds T_1.

We fill in T_1 Right-to-Left while scanning T Right-to-Left. We maintain the pointer *p* that points to the currently processed position in the array *positionsDistinctLMS* (processed Right-to-Left). If the currently processed position *i* in T corresponds to the starting position of LMS substring *s*, we compare *i* with the current position stored at *positionsDistinctLMS*[*p*]. If these positions are equal, then this means that the currently processed LMS substring *s* is either long or the representative of identical LMS substrings. This also means that in case *s* is a short LMS, then no other LMS the same as *s* has been processed yet. If *s* is a short LMS, we retrieve the integer-name of *s* stored at *namesDistinctLMS*[*p*] and place this name in the corresponding entry of the hash table using hash index equal to ~s (the complement of *s*). Either *s* is long or short, we fill in the current slot of T_1 with its integer-name stored at *namesDistinctLMS*[*p*]. In case, if the current starting position *i* in T corresponds to an LMS substring *s* whose position is not stored at *positionsDistinctLMS*, we extract *s* from T, and use the hash index ~*s* to retrieve its integer-name stored in the hash table at index ~s and place the name into the current slot of T_1. This procedure takes linear time.

4 Experimental Result

The major motivation for our algorithm was to time-efficiently construct an FM-index together with BWT for a large genome in practical space (less than 16GB of RAM). Taken this into account, to benchmark the performance of our algorithm, we chose the existing tools that build an FM-index or BWT for large genomes in practical space. We chose BWT-IS algorithm (implemented by the authors) because it can calculate the BWT for a human-size genome. We wrote our own script that given BWT-IS's output BWT, calculates an FM-index. The other chosen tool was Bowtie-2, [8] that builds an FM-index using algorithm by Kärkkäinen [6]. Bowtie-2 can build FM-index for small size genomes (less than or equal to 4Gb) and large genomes (greater than 4Gb). Finally, we show the results of BRAT-nova [4] that builds an FM-index exclusively for mapping bisulfite-treated reads to identify methylation, an important epigenetic marker.

We intentionally did not choose any of SA-IS algorithms because they run in space greater than 16GB and because publically available implementations of SA-IS have limitation to work with strings less than 2^{31}. We used BWT-IS algorithm that was kindly provided by the authors, and we used versions bowtie2-2.3.2 (for a human genome) and bowtie2-2.2.5 (for two strands of a human genome). All programs were compiled using the provided Makefiles. No additional options were used with Bowtie2.

We used total of three data sets. The first data set was the human genome GRCh38, from which we removed long runs of consecutive N characters leaving at most 49 of consecutive Ns. The size of this resulting genome was 2,934,896,319. The second data set was concatenation of two strands (positive and negative) of the same human genome. We took the reverse of the negative strand concatenated with the reverse of the positive strand. The reason for this choice of concatenation was the way the DNA reads are mapped to the FM-index: this way allows mapping reads starting with the starts of the reads that have the least number of sequencing errors. Finally, we used two strands of GRCh38 just as in the second step, but with all Cs converted to Ts. This index is used to map bisulfite-treated reads to identify methylation within a genome; in particular, this index is used in BRAT-nova, a mapping tool for bisulfite-treated reads. Hereafter, we will call these three data sets as *hg-one-strand*, *hg-two-strands* and *hg-two-strands-bs* respectively. Table 1 shows the running time of the tools on all three data sets measured in seconds and RAM usage. For BWT-IS, the running time is the sum of the time needed to run BWT-IS and the time to build the FM-index from the resulted BWT.

Bowtie-2 builds an FM-index for forward strand and reverse strands separately (first, it builds the FM-index for forward strand and then for the reverse strand). To make comparison fair, in Table 2 for Bowtie-2 we show time as reported by the program required to build the FM-index for forward strand only. For BWT-IS and our algorithm BWT-ISFM, we report the time as reported by the Linux command */usr/bin/time -v* by summing up *user* and *system* times, and for Bowtie-2 as reported by the tool. BWT-IS does not support strings of length greater than 2^{32}-1, so there are no results for data sets that use two strands of a human genome. To make our report complete, here we report time spent by BWT-IS to build BWT for *hg-one-strand*: it took 728sec. The rest 1536sec is required to build

Table 1: Time and RAM Usage

Tool	*hg-one-strand*		*hg-two-strands*		*hg-two-strands-bs*	
	Time	RAM	Time	RAM	Time	RAM
BWT-IS + FM-index	2264sec	5.45GB 1.86*n*	-	-	-	-
Bowtie2	3190sec	5.58GB 1.90*n*	7154sec	15.70GB 2.86*n*	7972sec	16.02GB 2.73*n*
BRAT-nova	-	-	-	-	38826sec	7.56GB 1.29*n*
BWT-ISFM	1397sec	6.15GB 2.10*n*	2736sec	13.00GB 2.22*n*	2429sec	11.47GB 1.95*n*

an FM-index. Our program BWT-ISFM shows the best time on hg-one-strand among the three tools compared. BWT-ISFM is 2.3, 2.6 and 3.3 times faster than Bowtie-2 on these three data sets; and it is 15.9 times faster than BRAT-nova. It shows comparable results with BWT-IS, but in addition our tool supports larger input strings.

All experiments were run on a 6-core Intel Xeon Processor 2.8GHz, 198GB RAM, and 216TB of raw storage space running Linux Ubuntu. RAM usage was measured as the maximum resident size. For BWT-IS we report memory usage while running BWT-IS. Space is reported in GB and as a function relative to the genome length *n*. For example, on *hg-one-strand*, BWT-IS used total of 5.45GB, which is 1.86*n* of bytes.

On *hg-one-strand*, our program uses slightly more memory than BWT-IS and Bowtie-2. On two strands of a human genome, BWT-ISFM shows better results than Bowtie-2 in terms of RAM. Compared to BRAT-nova, BWT-ISFM uses 1.5 times more space, but is 15.9 times faster. Overall, our program demonstrates a good practical tradeoff between space and time performance.

The source code for BWT-ISFM together with the scripts used in this benchmarking as well as the User Manual can be found at:

https://drive.google.com/drive/folders/0Bx79W9h8ZBHeZTlZRzFPSVVaRjQ

5 Conclusion and Future Improvements

In our work we extended BWT-IS algorithm to build an FM-index on fly while constructing BWT. We tuned our implementation to achieve a good balance between the running time and RAM usage while running on large genomes (of size at most 2^{32} - 1) as input. We think our algorithm can be extended to work with genomes of size up to 2^{34} (instead of up to 2^{33}) as long as the total number of LMS substrings in a given genome-string T fits into 32 bits. For the genome of size 2^{34}-1, this would mean that the total number of LMS substrings (i.e. the length of T_1) should be no more than *n*/4. In this case, our algorithm instead of using 1 bit to indicate whether a position is greater than MAXUI, will have to keep 2 most significant bits of the position.

6 Acknowledgement

The author thanks Stefano Lonardi and Tim Close (UC Riverside) for access to their computing server.

This work was supported in part by a 2017-1018 California State University Research, Scholarly, and Creative Activities Grant.

References

[1] Michael Burrows and David J. Wheeler. 1994. A Block Sorting Lossless Data Compression Algorithm. Technical Report 124, Digital Equipment Corporation.

[2] Jasbir Dhaliwal, Simon. J. Puglisi, and Andrew Turpin. 2012. Trends in suffix sorting: a survey of low memory algorithms. In *Proceedings of the Thirty-fifth Australasian Computer Science Conference* (ACSC '12), Mark Reynolds and Bruce Thomas (Eds.), Vol. 122. Australian Computer Society, Inc., Darlinghurst, Australia, Australia, 91-98.

[3] Paolo Ferragina and Giovanni Manzini. 2000. Opportunistic data structures with applications. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, Redondo Beach, CA, 2000, 390-398.

[4] Elena Y. Harris, Rachid Ounit, and Stefano Lonardi. 2016. BRAT-nova: Fast and accurate mapping of bisulfite-treated reads. *Bioinformatics* 32, 17 (Feb 2016), 2696–2698.

[5] Steve Hoffmann, Christian Otto, Stefan Kurtz, Cynthia M. Sharma, Phillipp Khaitovich, Jörg Vogel, Peter F. Stadler, and Jörg Hackermüller, 2009. Fast mapping of short sequences with mismatches, insertions and deletions using index structures. *PLoS Computational Biology* 5, 9 (Sep 2009), e1000502.

[6] Juha Kärkkäinen. 2007. Fast bwt in small space by blockwise suffix sorting. *Theoretical Computer Science* 387, 3 (Nov 2007), 249-257.

[7] Pang Ko and Srinivas Aluru. 2005. Space efficient linear time construction of suffix arrays. *Journal of Discrete Algorithms* 3, 2-4 (June 2005) 143-156.

[8] Ben Langmead, Cole Trapnell, Mihai Pop, and Steven L. Salzberg. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biology* 10, 3 (Mar 2009), R25.

[9] Udi Manber and Gene Myers. 1990. Suffix arrays: a new method for on-line string searches. In *Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms* (SODA '90). Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 319-327.

[10] Ge Nong, Sen Zhang, Wai H. Chan. 2009. Linear suffix array construction by almost pure induced-sorting. In *Data Compression Conference* (DCC). IEEE Computer Society, Snowbird, UT, USA, 193–202.

[11] Daisuke Okanohara and Kunihiko Sadakane. 2009. A Linear-Time Burrows-Wheeler Transform Using Induced Sorting. In *Proceedings of the String Processing and Information Retrieval: 16th International Symposium* (SPIRE 2009). Saariselkä, Finland, 90–101.

Global Optimization Approach for Circular and Chloroplast Genome Assembly

Sebastien François [1] Rumen Andonov [1*] Dominique Lavenier [1]
Hristo Djidjev [2]
[1] Univ Rennes, Inria, CNRS, IRISA, F-35000 Rennes, France
[2] Los Alamos National Laboratory, Los Alamos, NM 87545, USA

Abstract

We describe a global optimization approach for genome assembly where the steps of scaffolding, gap-filling, and scaffold extension are simultaneously solved in the framework of a common objective function. The approach is based on integer programming model for solving genome scaffolding as a problem of finding a long simple path in a specific graph that satisfies additional constraints encoding the insert-size information. The optimal solution of this problem allows one to obtain new kind of contigs that we call distance-based contig. We test the algorithm on a benchmark of chloroplasts and compare the quality of the results with recent scaffolders.

keywords: genome assembly, scaffolding, unitig, contig, longest simple weighted path problem, integer programming

1 Introduction

Modern Next-Generation Sequencing (NGS) techniques output billions of short DNA sequences, called *reads*, and the typical way to process this information is by using *de novo* assembly. However, assembling these fragmented raw data into complete genomes remains a challenging computational task. This is a very complex procedure, usually involving three main steps: (1) generation of *contigs*, which are contiguous genomic fragments issued from the overlapping of the reads; (2) constructing *scaffolds*–set of ordered and oriented contigs along the genome interspaced with gaps; (3) *finishing*, which aims to complete the assembly by inserting DNA text in the gaps between the ordered contigs.

The first step generates a list of contigs that usually represent the "easily assembled regions" of the genome. Building contigs is currently supported by methods using a specific data structure called *de-Bruijn* graph [13]. Here we use *unitigs*–a special kind of high-confidence contigs that represent maximal unambiguous paths in the de-Bruijn graphs. Despite the progress done in the domain, complex regions of the genome (e.g., regions with many repeats) generally fail to be assembled by these techniques. If the genome contains repeats longer than the size of the reads, the entire genome cannot be built in a unique way.

Whereas the main challenge of the first step relies on handling huge volume of data, the scaffolding step manipulates data of moderate size. However, the problem remains largely open because of its NP-hard complexity [9]). The goal here is to provide a reliable order and orientation of the contigs in order to link them together into *scaffolds*. Contigs can be linked together using *paired-end* or *mate-pair* reads [16, 11]. This complementary data is due to the ability of the sequencing technology to provide couples of reads that are separated by a known distance (called *insert size*). They bring a long distance information that is not used in the first assembly stage, but is essential for the second.

The scaffolding phase usually produces multiple scaffolds. Moreover, these scaffolds may contain regions that have not been completely predicted. Hence, two additional steps, *gap-filling* and *scaffold extension* (elongating and concatenating the contigs after the scaffolding step) are typically needed to complete the genome.

The strategy proposed here differs significantly from the approaches described in the literature. While the latter apply various heuristics for tackling the different assembly stages one after another separately, our methodology consists of developing a global optimization approach where the scaffolding, gap-filling, and scaffold extension steps are simultaneously solved in the framework of a common objective function. Our approach is based on integer programming models for solving the genome scaffolding as a problem of finding a long simple path in a specific graph that satisfies as many as possible of the additional constraints encoding

*Corresponding author. Email: randonov@irisa.fr

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

the insert-size information [4].

We are not aware of previous approaches on scaffolding based on longest path problem reduction. Most previous work on scaffolding is heuristics based, e.g., SSPACE [2], GRASS [6], BESST [15] and SPAdes [1]. Such tools may find in some cases good solutions, but their accuracies cannot be guaranteed or predicted. Exact algorithms for the scaffolding problem are presented in [17], but the focus of that work is on finding structural properties of the contig graph that will make the optimization problem of polynomial complexity. In [12], integer linear programming is used to model the scaffolding problem, with an objective to maximize the number of links that are satisfied. In order to avoid sub-cycles in the solution, the authors use an incremental process, where cycles that may have been produced by the solver are forbidden in the next iteration. Integrating the distances between contigs and accounting for possible multiplicities of the contigs (repeats, copy-counts) is indicated as future improvement in [12], while it has been realized in our approach.

This paper focuses on circular genomes and, in particular, on chloroplasts. The reasons for this choice are as follows. Chloroplasts possess circular and relatively small genomes. The particularity of these genomes is the presence of numerous repetitions, while these are the main chalenges for the modern genome assembly techniques. On the other hand, the size of the chloroplast genome permits assembling them rapidly (each one of the instances from the considered benchmark except one, EuglenaGracilis genome, has been solved for less that 1 sec.) and so we were able to refine our strategy and to focus entirely on the quality of the obtained results.

The contributions of this study are as follows:

- We adapt and further develop the general case approach proposed in [4] to the case of circular genomes. Using the specificities of this particular case we succeed to simplify significantly the sophisticated mixed integer linear program (MILP) described in [4].

- We propose an exact approach for scaffolding in the case of circular genomes as a problem of finding longest paths in specific unitig graphs with additional set of constraint distances between couples of vertices along these paths.

- We deeply analyze the reasons for the existence of a huge number of multiple equivalent optimal solutions. These solutions are mainly explained by the presence of repetitions in the set of unitigs. We find sufficient conditions for the existence of multiple solutions zones and propose an algorithm for identifying these zones.

- By using the optimal path found by the MILP model, our algorithm permits merging a set of unitigs satisfying the link distances into what we call *distance-based contigs*. These contigs, together with the other unitigs, are given to QUAST [7] for assessment.

- We tested this strategy on a set of 33 chloroplast genome data and compared the results with some of the most recent scaffolders (namely with SPAdes [1], SSPACE [2], BESST [15] and SWALO [14]).

- Our numerical experiments show that our approach produces assemblies of higher quality than the above heuristics on the considered benchmark.

2 Modeling the scaffolding problem

In this section we adapt the optimization approach proposed in [4] to the particularities and characteristics of the circular genomes. Section 2.1 describes the graph modeling that is common for both approaches, while the mathematical programming formulation presented in section 2.2 includes enhancements of the model that, while making it less general, greatly increase its efficiency for chloroplast genome scaffolding.

2.1 Graph Modeling

The input data for our approach are the following:

- A set of *unitigs* together with their *copy-count* (multiplicity). Only unitigs larger than a predefined threshold (cf section 4.1) are considered. The copy-count is determined from k-mer counting techniques (cf section 4.1.2).

- A list of overlaps between the unitigs. Two unitigs overlap if they share a minimum of common nucleotides at their extremities.

- A list of oriented couples of unitigs (links). Links are determined from *paired-end* or *mate-pair* information. Due to insert size fluctuation, an interval distance is associated with any link from this list.

We follow the modeling from [4] where the scaffolding problem is reduced to a path finding in a directed graph $G = (V, E)$, called a unitig graph, where both vertices V and edges E are weighted. The set of vertices V is

generated based on the set C of the unitigs according the following rules: the unitig i is represented by at least two vertices v_i and v_i' (forward/inverse orientation respectively). If the unitig i is repeated k_i times (this value corresponds to the copy-count), it generates a set $\mathcal{C}_i$ of $2k_i$ vertices. If two different vertices v and w belong to $\mathcal{C}_i$ and have the same orientations, we can use the notation $v \approx w$. Let us denote $N = \sum_{i \in C} k_i$; thus $|V| = 2N$.

The edges are generated following given patterns—a set of known overlaps/distances between the unitigs. Any edge is given in the graph G in its forward/inverse orientation. We denote by e_{ij} the edge joining vertices v_i and v_j and the inverse of edge e_{ij} by $e_{j'i'}$. Let w_v be the length of the unitig corresponding to vertex v and denote $W = \sum_{v \in V} w_v$. Moreover, let the weight l_e on the edge $e = (v_i, v_j)$ correspond to the value of the overlap/distance between unitigs represented by v_i and v_j. The problem then is to find a path in the graph G such that the total length (the sum over the traversed vertices and edges) is maximized, while a set of additional constraints is also satisfied:

- For any i, either vertex v_i or v_i' is visited (participates in the path).
- The orientations of the nodes does not contradict the constraints imposed by the links. This is at least partially enforced by the construction of G.

To any edge $e \in E$ we associate a variable x_e. Its value is set to 1, if the corresponding edge participates in the assembled genome sequence (the associated path in our case), otherwise its value is set to 0. There are two kinds of edges: edges corresponding to overlaps between unitigs, denote them by O (from overlaps), and edges associated with the links relationships, denote them by L. We therefore have $E = L \cup O$. Let l_e be the length assigned to the edge $e = (u, v)$. We define l_e $\forall e \in O$ such that $l_e < 0$ and $|l_e| < \min\{w_u, w_v\}$ is the overlap between the contigs corresponding to v_i and v_j, and $l_e > 0$ $\forall e \in L$, where l_e is the link distance between unitigs represented by v_i and v_j.

Let $\delta^+(v) \subset E$ (resp. $\delta^-(v) \subset E$) denote the sets of edges outgoing from (resp. incoming to) v.

2.2 Mixed Integer Linear Programming Formulation

The crucial observation in the approach proposed in [4] is that the genome can be assembled by searching for a particular longest path in the associated unitig graph. However, the beginning and the end of this path are unknown in the general case. This constraint leads to the sophisticated model described in [4]. Here we use the following facts/hypotheses for chloroplast genomes in order to simplify the above general approach:

(1) Chloroplast genomes are circular;

(2) We assume that any input unitig is part of the genome.

(3) We assume that the entire genome is sufficiently covered (no gaps in its sequence).

In our runs we choose the largest unitig (say s) to play the role of the beginning and the end of the genome. Consequently, we introduce a supplementary vertex t that gets all incoming edges from s. Specifically, each edge (x, s) we replace by an edge (x, t) and set $\delta^-(t) = \delta^-(s)$, $\delta^+(t) = \emptyset$, and $\delta^-(s) = \emptyset$. Vertices s and t will be considered respectively as the source (start) and the sink (end) of the path we are looking for.

Furthermore, to any vertex $v \in V \setminus \{s\}$ we associate the variable i_v s.t.

$$0 \leq i_v \leq 1 \tag{1}$$

encoding whether v is in the solution path. Moreover, each vertex (or its inverse) should be visited at most once, which we encode as

$$\forall (v, v') : i_v + i_{v'} \leq 1. \tag{2}$$

We associate a binary variable for any edge of the graph, i.e.,

$$\forall e \in O : x_e \in \{0, 1\} \text{ and } \forall e \in L : g_e \in \{0, 1\}. \tag{3}$$

The two possibles states for a vertex v (to be (or not) an intermediate vertex in the path) are enforced by the following constraints

$$i_v = \sum_{e \in \delta^+(v)} x_e = \sum_{e \in \delta^-(v)} x_e. \tag{4}$$

It is then obvious that the real variables $i_v, \forall v \in V$ take binary values.

We introduce a continuous variable $f_e \in R^+$ to express the quantity of the flow circulating along the edge $e \in E$. Without this variable, the solution found may contains some loops and hence may not be a simple path. We put a requirement that no flow can use an edge e when $x_e = 0$, which can be encoded as

$$\forall e \in E : 0 \leq f_e \leq W x_e, \tag{5}$$

where W is as defined above ($W = \sum_{v \in V} w_v$).

We use the flows f_e in the following constraints, $\forall v \in V \setminus \{s\}$,

$$\sum_{e \in \delta^-(v)} f_e - \sum_{e \in \delta^+(v)} f_e = i_v (w_v + \sum_{e \in \delta^-(v)} l_e x_e), \tag{6}$$

while for the source vertex we require

$$\sum_{e \in \delta^+(s)} f_e = W. \tag{7}$$

We furthermore observe that, because of (4), the constraint (6) can be written as follows

$$\forall v \in V : \tag{8}$$
$$\sum_{e \in \delta^-(v)} f_e - \sum_{e \in \delta^+(v)} f_e = \; i_v w_v + \textstyle\sum_{e \in \delta^-(v)} l_e x_e.$$

The constraint (8) is linear and we keep it in our model instead of (6).

The model so far defines a solution to the longest path problem. We need also to add information related to the links distances. For that reason, we associate a binary variable g_e with each link e. For $(u, v) \in L$, the value of $g_{(u,v)}$ is set to 1 only if both vertices u and v belong to the selected path and the length of the considered path between them is in the given interval $[\underline{L}_{(u,v)}, \overline{L}_{(u,v)}]$. Constraints related to links are :

$$g_{(u,v)} \leq i_u \text{ and } g_{(u,v)} \leq i_v \tag{9}$$

$$\forall (u, v) \in L : \tag{10}$$
$$\sum_{e \in \delta^+(u)} f_e - \sum_{e \in \delta^-(v)} f_e \geq \; \underline{L}_{(u,v)} g_{(u,v)} - M(1 - g_{(u,v)}),$$

$$\forall (u, v) \in L : \tag{11}$$
$$\sum_{e \in \delta^+(u)} f_e - \sum_{e \in \delta^-(v)} f_e \leq \; \overline{L}_{(u,v)} g_{(u,v)} + M(1 - g_{(u,v)}),$$

where M is some big constant.

Our goal is to find a long path in the graph such that as many as possible link distances are satisfied. The corresponding objective function hence is of the form

$$\max \left(\sum_{e \in O} x_e l_e + \sum_{v \in V} w_v i_v + p \sum_{e \in L} g_e \right) \tag{12}$$

where p is a parameter to be chosen as appropriate (currently $p = 1$).

3 Dealing with multiple optimal solutions

By its nature, the information provided by the overlaps and mate pairs is not always sufficient to determine the assembly in a unique way. For instance, the unitig graph G is symmetric by construction, e.g., if there is an edge (v, w) between vertices v and w, then there is an edge (w', v') between their inverses w' of w and v' of v. Moreover, it contains repeated identical unitigs, which are modeled by different vertices of G. For all the above reasons, for each optimal solution (path) p^* found by our algorithm, there are typically multiple (exponential in the worst case) number of equivalent solutions (paths). Such paths are different from p^* as sequences of vertices of G, but correspond to the same set of unitigs (and their inverted copies) and satisfy the same number of links, and hence are equally "optimal" from the point of view of the optimization problem (1)–(12). This issue is especially pronounced for chloroplasts due to their higher number of repeated/symmetrical regions.

Choosing just any arbitrary path from the set of equivalent optimal ones can result into an assembly different from the genome reference, which is the main criterion for evaluating the accuracy of the prediction. Therefore, our strategy is to detect in the optimal path multiple solution portions and to separate them from subpaths that cannot be replaced by equivalent ones. This latter type of subpaths will be merged in what we call *db-contigs* (distance-based, i.e., contiguous sequences that satisfy the link distances). Obviously, none of the optimal solutions is eliminated while proceeding in such a manner. We call these zones "unsafe" and "safe," respectively.

In this section we describe a method to decompose a solution path into safe and unsafe zones. Our algorithm is heuristic, meaning that it does not necessarily identify all safe zones, but, as our experiments show, it works well in practice.

Formally, we call two paths $p_1 = (v_1, \ldots, v_k)$ and $p_2 = (w_1, \ldots, w_k)$ of G *equivalent*, if they satisfy the same set of links and their components are permutations of the same set of unitigs (and their inverted copies). These paths can differ (or not) as sequences of base pairs. If p is a path in the unitig graph representing a solution of the optimization problem, we call a subpath p' of p a *safe zone* of p if there exists no path in the graph G minus $p \backslash p'$ that is equivalent to and different from p', and p' is a maximal subpath with this property. Safe zones are in fact subpaths containing a number of satisfied links, since each such link adds a constraint that reduces the number of subpaths that may be equivalent to it. Removing all safe zones from p leaves a set of paths that we call *unsafe zones*. We call a path p *link-closed* if for any link that has as an endpoint an intermediate vertex of p, its other endpoint is also p.

Next, we will illustrate a method for identifying unsafe zones by an example. Consider a unitig v_s of multiplicity two. According to the graph-generation rules, there are vertices v_{s0} and v_{s1} in G corresponding to v_s in

the forward orientation and their corresponding vertices v'_{s0} and v'_{s1} in the opposite direction. Assume also that there exists a link-closed subpath $p = (v_k, v_{k+1}, \ldots, v_r)$ of a solution to the optimization problem such that $v_k = v_{s0}$ and $v_r = v'_{s1}$. Remember that, for each edge (v_i, v_{i+1}) from p, the inverse edge $(v'_{i+1}, v'_i,)$ also exists in the unitig graph. Then we show that the inverse of p, i.e. the path $p' = \text{inv}(p) = (v'_r, v'_{r-1}, \ldots, v'_k)$ of inverted unitigs is also an optimal solution of the optimization problem. Obviously, $length(p) = length(p')$. Since $v'_r = v_{s1} = v_{s0}$ and $v'_k = v'_{s0} = v'_{s1}$, the paths p and p' have the same sets of unitigs corresponding to their vertices and have identical unitigs at the beginning and their ends, but they are different as paths (sequences of vertices). The subsequence $p = (v_{k+1}, \ldots, v_{r-1})$ is in this sense unsafe zone in the solution path. An example of such unsafe zone is illustrated on Figures 1.

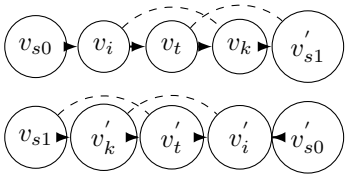


Figure 1: **Top:** a path p containing two links visualized with dashed lines; **Bottom:** its reversible path p'. Note that v_{s0}(resp. v'_{s0}) is identical to v_{s1}(resp. v'_{s1}).

It turns out that the type of subpath illustrated in the previous example is quite common and most of the unsafe zones that we have identified in our experiments can be captured using it. The algorithm for safe/unsafe zones detection based on using this pattern works as follows:

(1) The vertices belonging to any satisfied link from the optimal path p^* found by the model in section 2.2 are considered elements of a potential db-contig.

(2) Potential db-contigs that overlap at least one vertex are merged in new (longer) potential db-contigs.

(3) Any vertex outside the potential db-contigs is considered as unsafe.

(4) For any potential db-contig C we apply the following algorithm.

  (a) Any vertex $v_s \in C$ is initialized as safe.

  (b) For any safe vertex $v_s \in C$ with multiplicity of at least two, and such that exists a couple (v_{s0}, v'_{s1}) belonging to C, and such that the subpath between v_{s0} and v'_{s1} is link-closed do: (i) indicate as unsafe both vertices v_{s0} and v'_{s1}; (ii) indicate the path between v_{s0} and v_{s1} as a new potential db-contig.

(5) All adjacent safe vertices are merged in true db-contigs (new meta-vertices).

The algorithm is illustrated on Figures 2, 3 and 4.

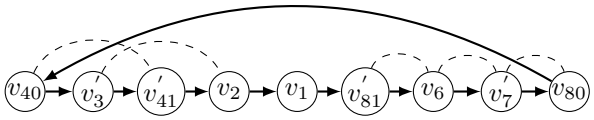


Figure 2: The initial solution.

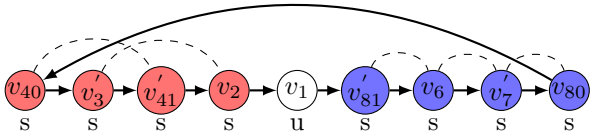


Figure 3: Steps 1, 2 and 3. Two potential db-contigs are created (the first one is red colored, the second is blue colored). Their vertices are initially labeled as safe. The vertex v_1 is labeled as unsafe since it is outside the potential db-contigs.

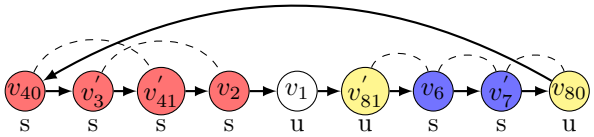


Figure 4: Step 4. Two repetitions are detected : the couples (v_{40}, v'_{41}) and (v'_{81}, v_{80}). However, the path (v_{40}, v'_3, v'_{41}) is not reversible, since it is not link-closed (because of the link (v'_3, v_2)). On the other hand, the path $(v'_{81}, v_6, v'_7, v_{80})$ is reversible. The vertices v'_{81} and v_{80} are labeled as unsafe. Finally, two true db-contigs are created : the first one, C_1 (in red), contains the subpath $(v_{40}, v'_3, v'_{41}, v_2)$, the second one C_2 (in blue), contains the subpath (v_6, v'_7). These two db-contigs, together with vertices/unitigs v_2 (in white) and v_{80} (in yellow) are given for assessment to QUAST.

In order to evaluate the quality of obtained solution we use QUAST [7]. Note that this tool requires for input just a set of contigs without indication for their repetition and orientation (for example, the

input concerning the instance from Figure 4 consists in contigs C_1, C_2, v_1 and v_{80} uniquely). QUAST maps any of them to the reference genome on order to assess its quality.

Note that this algorithm does not necessarily find all unsafe/safe zones, but it works well in practice. Correctly identifying all such zones is an interesting research problem, whose solution can further improve the quality of our tool. In the next section we report some experimental results comparing our tool with some of the best existing similar tools.

4 Experimental Analysis

4.1 Data Generation

4.1.1 Simulated Data

From 33 chloroplast reference genomes (cf Table 1), 33 datasets of mate-pairs or pair-ended reads are generated with the art-illumina software with 100x depth of coverage [8]. For each dataset, the two following tasks are performed: (i) unitig generation; (ii) link computation.

4.1.2 Unitig generation

Unitigs are generated with the Minia assembler [3]. A range of different k-mer sizes are tried to find the one that yields the best assembly.

For each unitig, its abundance (copy-count) is computed, that is, the number of times it appears in the genome. For that, we define the kmer abundance as the number of times this kmer or its reverse-complement appears in the read files. The abundance of a unitig is then computed as the average abundance of all its kmers. This abundance is computed and returned by the Minia software.

In theory, the abundance of a unitig that is not repeated in the genome should be equal to the depth of coverage of sequencing, twice that amount for duplicated unitigs, and so on. We assume that the longest unitig is not duplicated, i.e. that its abundance is equal to the depth of coverage. The multiplicity of each unitig is then simply computed as its abundance divided by the depth of coverage, rounded to the nearest upper integer value.

This strategy provides an estimation of the coverage, but its accuracy strongly depends of the length of the unitigs. Longer the unitigs, better the estimation. Actually, for very short unitigs, we can only provide intervals of confidence or, at least an upper bound.

4.1.3 Link computation

Each mate-pair or pair-ended read is individually mapped to unitigs with minimap [10]. We discard reads that map ambiguously to several locations. Reads of a pair that map to different unitigs indicate a mate-pair link in the graph. To avoid false positives, we only keep links that are validated by at least 5 pairs. The link size is estimated thanks to the known inserts size and mapping position in each unitig, and averaged over all pairs that confirm the link.

4.1.4 Computational results

We have generated a data set of 33 chloroplasts genomes obtained from the NCBI website (https://www.ncbi.nlm.nih.gov/genome). In order to simulate mate-pairs and pair-ends sequencing, we used the ART simulator Illumina [8].We have produced reads with a length of 250bp and 100X coverage. Two types of simulation were performed: for the pair-end simulation the inserts size was 600bp, while, for the mate-pairs simulation, we used an insert of 8000bp. The reads were subsequently assembled in unitigs by Minia [3] and the generated *fasta* file was input to the scaffolders that needed it (SPAdes [1] and SWALO [14] work directly with the reads and do not require it). The unitigs were produced with an abundance of 4 and a k-mer of 125.

The assemblies were evaluated by QUAST [7] tool by comparison with the reference genome that was used for the simulation. (More detailed experimental data is given in the Appendix.) Our tool is denoted by GAT (Genscale Assembly Tool). In our experiments, GAT has been as good as, and often better, than the best current scaffolding tools, while ensuring good coverage of the reference genome (a parameter that tends to degrade with other scaffolders). It has been particularly good in case of pair-ends computations by ensuring a regular and nearly optimal assembly.

During the mate pairs computations GAT performed well by producing often the smallest number of contigs and was outperformed only in the case of Atropa genome. GAT produces on average fewer contigs than its competitors. Moreover, it ensures the best genome coverage. This indicates that the output produced by our tool are reliable, complete, and don't lose information compared to the original genome. SWALO failed to assemble 10 genomes out of 33, SSPACE 3 genomes, and BESST–one genome. SPAdes and GAT where the only tools for which QUAST did not indicate any missassemblies (these results are given in the extended version of this paper [5]).

In the case of pair-end simulations, we have obtained

equally good results. The performance of GAT and SPAdes are very close in term of average number of contigs (cf. Figure 5). However, SPAdes is outperformed by GAT, BESST and SWALO concerning the genome coverage (cf. Figure 6). On this figure we also observe that BESST is as reliable as GAT, but it couldn't solve Euglena (21)–something that GAT achieved.

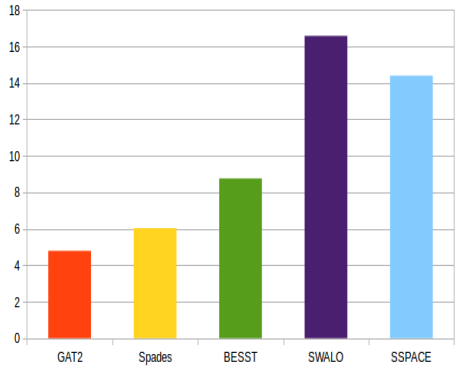


Figure 5: Pair-ends data : Average number of contigs comparison.

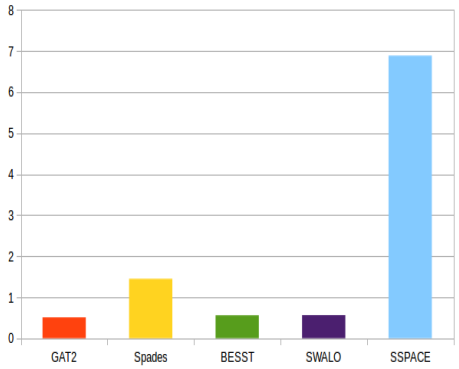


Figure 6: Pair-ends data : Average fraction of genome left out comparison.

5 Conclusion

Here we design and test an algorithm for scaffolding and gap filling phases in the case of circular genomes. Our approach is based on a version of the longest path problem solved by MILP modeling. It works both in case of mate-pairs and pair-ends distances. On a benchmark of 33 chloroplast genomes our algorithm significantly outperforms four recent scaffolding heuris-

No	Genomes	Size	$\|V\|$	$\|O\|$	$\|L\|$	NSL
1	Acorus Calamus	153821	8	16	16	3
2	AdiantumCapillus Veneris	150568	20	24	24	5
3	Agrostis Stolonifera	136584	20	52	24	6
4	Angiopteris Evecta	153901	34	78	70	12
5	Anthoceros Formosae	161162	16	32	24	5
6	Arabidopsis Thaliana	161162	20	40	32	7
7	Arabishirsuta	153689	12	24	24	5
8	Atropa	156687	46	90	34	9
9	Capsella Bursa Pastoris	154490	12	24	24	5
10	Chaetosphaeridium Globosum	131183	8	16	16	3
11	Chara Vulgaris	184933	24	56	24	7
12	Chlorella Vulgaris	150613	52	50	50	24
13	Chlorokybus Atmophyticus	152229	10	18	18	4
14	Citrus Sinensis	160129	12	24	24	5
15	Cyanidioschyzon Merolae	149067	72	82	46	22
16	Cyanidium Caldarium	164921	38	36	32	15
17	Daucus Carota	155911	8	16	16	3
18	Draba Nemorosa	153289	12	24	24	5
19	Eimeria Tenella	160604	10	18	18	4
20	Epifagus Virginiana	70028	12	24	24	5
21	Euglena Gracilis	143171	146	554	30	5
22	Gossypium Barbadense	160317	12	24	24	5
23	Gossypium Hirsutum	160301	14	28	24	5
24	Gracilaria Tenuistipitata	183883	54	54	44	21
25	Guillardia Theta	121524	44	88	24	5
26	Helianthus Annuus	151104	10	18	18	4
27	Huperzia Lucidula	154259	20	48	20	5
28	Lactuca Sativa	152765	8	16	16	3
29	Lepidium Virginicum	154743	24	48	48	11
30	Liriodendron Tulipifera	159886	8	16	16	3
31	Lobularia Maritima	152659	16	32	32	7
32	Lotus Corniculatus	150519	20	80	32	7
33	Pinus	116864	58	128	12	6

Table 1: The benchmark containing 36 chloroplast genomes whose names given in the first column. The second column contains their lengths. We observed that this value equals the value given by the first term of the objective function (12). The third and fourth columns give the size of the graph (i.e. number of vertices and edges). $|L|$ indicates the number of given links, while NSL stands for number of satisfied links in the solution.

tics with respect to the quality of the scaffolds. The obtained results fully justify the efforts for designing exact approaches for genome assembly. Regardless of that, we consider the current results as a work in progress. The biggest challenge is to extend the method to much bigger genomes. We are currently implementing advanced combinatorial optimization decomposition techniques to increase the scalability of the approach without sacrificing the accuracy of the results.

Acknowledgments

We would like to thank Guillaume Rizk for adapting the Minia assembler [3] for the purpose of our algorithm and for his help in data generation. Many thanks to Rayan Chikhi for valuable discussions.

This work has been supported in part by Inria international program Hipcogen.

References

[1] Anton Bankevich, Sergey Nurk, Dmitry Antipov, Alexey A Gurevich, Mikhail Dvorkin, Alexander S Kulikov, Valery M Lesin, Sergey I Nikolenko, Son Pham, Andrey D Prjibelski, Alexey V Pyshkin, Alexander V Sirotkin, Nikolay Vyahhi, Glenn Tesler, Max A Alekseyev, and Pavel A Pevzner. Spades: a new genome assembly algorithm and its applications to single-cell sequencing. *Journal of computational biology : a journal of computational molecular cell biology*, 19(5):455477, May 2012.

[2] Marten Boetzer, Christiaan V. Henkel, Hans J. Jansen, Derek Butler, and Walter Pirovano. Scaffolding pre-assembled contigs using SSPACE. *Bioinformatics (Oxford, England)*, 27(4):578–579, February 2011.

[3] Rayan Chikhi and Guillaume Rizk. Space-efficient and exact de bruijn graph representation based on a bloom filter. *Algorithms for Molecular Biology*, 8(1):22, 2013.

[4] S. François, R. Andonov, H. Djidjev, and D. Lavenier. Global optimization methods for genome scaffolding. In *8th International Network Optimization Conference (INOC)*, 2017. to appear in the special issue of Electronic Notes in Discrete Mathematics (ENDM) V. 64.

[5] S. François, R. Andonov, D. Lavenier, and H. Djidjev. Global optimization approach for circular and chloroplast genome assembly. *bioRxiv*, 2017.

[6] Alexey A. Gritsenko, Jurgen F. Nijkamp, Marcel J.T. Reinders, and Dick de Ridder. GRASS: a generic algorithm for scaffolding next-generation sequencing assemblies. *Bioinformatics*, 28(11):1429–1437, 2012.

[7] Alexey Gurevich, Vladislav Saveliev, Nikolay Vyahhi, and Glenn Tesler. Quast: quality assessment tool for genome assemblies. *Bioinformatics*, 29(8):1072–1075, 2013.

[8] Weichun Huang, Leping Li, Jason R Myers, and Gabor T Marth. Art: a next-generation sequencing read simulator. *Bioinformatics*, 28(4):593–594, 2011.

[9] Daniel H. Huson, Knut Reinert, and Eugene W. Myers. The greedy path-merging algorithm for contig scaffolding. *J. ACM*, 49(5):603–615, 2002.

[10] Heng Li. Minimap and miniasm: fast mapping and de novo assembly for noisy long sequences. *Bioinformatics*, 32(14):2103–2110, 2016.

[11] Paul Medvedev, Son Pham, Mark Chaisson, Glenn Tesler, and Pavel Pevzner. Paired de Bruijn graphs: A novel approach for incorporating mate pair information into genome assemblers. *Journal of Computational Biology*, 18(11):1625–1634, 11 2011.

[12] Briot Nicolas, Chateau Annie, Rémi Coletta, Simon de Givry, Philippe Leleux, and Schiex Thomas. An integer linear programming approach for genome scaffolding. In *Workshop on Constraint based Methods for Bioinformatics*, 2015.

[13] Pavel A. Pevzner, Haixu Tang, and Michael S. Waterman. An Eulerian path approach to DNA fragment assembly. *PNAS*, 98(17):9748–9753, 2001.

[14] Atif Rahman and Lior Pachter. Swalo: scaffolding with assembly likelihood optimization. *bioRxiv*, 2016.

[15] Kristoffer Sahlin, Francesco Vezzi, Björn Nystedt, Joakim Lundeberg, and Lars Arvestad. BESST - efficient scaffolding of large fragmented assemblies. *BMC Bioinformatics*, 15:281, 2014.

[16] James L. Weber and Eugene W. Myers. Human whole-genome shotgun sequencing. *Genome Research*, 7(5):401–409, 1997.

[17] Mathias Weller, Annie Chateau, and Rodolphe Giroudeau. Exact approaches for scaffolding. *BMC bioinformatics*, 16(Suppl 14):S2, 2015.

Adjusted Likelihood-Ratio Test for Variants with Unknown Genotypes

Ronald J. Nowling and Scott J. Emrich
Computer Science & Engineering
University of Notre Dame
Notre Dame, IN 46656
(rnowling,semrich)@nd.edu

Abstract

Association tests performed with the Likelihood-Ratio Test (LR Test) can be an alternative to F_{ST}, which is often used in population genetics to find variants of interest. Because the LR Test has several properties that could make it preferable to F_{ST}, we propose a novel approach for modeling unknown genotypes in highly-similar species. To show the effectiveness of this LR Test approach, we apply it to single-nucleotide polymorphisms (SNPs) associated with the recent speciation of the malaria vectors *Anopheles gambiae* and *Anopheles coluzzii* and compare to F_{ST}.

1 Introduction

Fixation index, or F_{ST}, has been used extensively in population genetics analyses (see [5, 10, 15] for insect-focused studies). F_{ST} is a score between 0 and 1 calculated from population frequencies of known alleles. To identify variants for further analysis, researchers often calculate F_{ST} for each single nucleotide polymorphism (SNP) individually, average individual F_{ST} scores over larger regions (windows), rank them using these scores, and then select interesting SNPs or regions based on an arbitrarily-chosen cutoff (e.g., top 500 or top 0.1%).

An alternative approach is performing Likelihood-Ratio Tests (LR Tests) using Logistic Regression (LogReg) models [2]. For each SNP, a LogReg model is trained, and then a LR Test is performed between the LogReg model and a null model based on the class probabilities [7]. LR Tests report p-values that can be used to identify statistically-significant variants relative to this null model. Note that LR Tests have been used extensively in human genome-wide association studies (GWAS) [1].

Population analysis of heterogeneous insect genomes often faces two challenges: small sample sizes and unknown genotypes. Because F_{ST} does not take samples sizes into account, the same F_{ST} score could be reported with 2 or 100 samples, as long as the observed frequencies of the alleles are identical. In contrast, LR Tests can account for sample sizes when determining the p-value of a SNP, which helps control type I errors (false positives).

Another concern is unknown genotypes that result from a variety of challenges, both biological (i.e., high levels of heterozygosity) and experimental (i.e., lower sampling coverage than expected). In humans and other organisms, unknown genotypes are often imputed using tools such as IMPUTE2 [8, 11] before performing single SNP association tests using tools such as SNPTEST [12]. Unknown genotypes in insect genomes, however, are rarely imputed because of the difficulty in doing so accurately with limited samples.

Rather than imputing unknown genotypes, we propose a framework that handles unknown genotypes directly. We make the conservative (uninformative) assumption that each unknown genotype has an equal probability of being each genotype. We then ensure that this assumption is reflected in the conditional class probabilities calculated by the LogReg models (Section 2.3). Then, in Section 3.2, we validate these resulting LogReg models by comparing predicted probabilities to analytically-calculated probabilities.

In Section 3.3, we compare the properties of F_{ST} and our LR Test approach using simulated data. We demonstrate that the p-values computed by the LR Test vary with the number of unknown genotypes and underlying sample sizes, while the F_{ST} scores do not.

As a specific example of a real-world application, we apply our LR Test framework to ≈1.7 million SNPs from the recently speciated malaria vectors *Anopheles gambiae* and *Anopheles coluzzii* from [5]. These data derive from a single chromosome arm (2L) containing relatively strong regions of differentiation [10, 15]. Identifying specific sequence-based differences is highly valuable for molecularly characterizing such closely-related species and ultimately to help understand speciation in these model systems [13]. Even though PCA analysis of samples from the two species has shown strong evidence for strong similarity within species

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

and clear differences between species [15], localizing key variants is ongoing work [14]. At a significance level of 1%, we find that as many as 522 positions on chromosome arm 2L are statistically significant after correcting for multiple comparisons. Of 1,633 positions with the highest possible F_{ST} score (1), only twenty overlap with this set of 522 significant positions.

This result suggests that the adjusted LR Test may be more specific than averaging SNP F_{ST} values across larger windows as performed by [10] and can better address unknown and heterogeneous genotypes than F_{ST} alone. We provide a reference implementation using scikit-learn in Asaph, a variant analysis toolkit. Note that since this framework uses common methods, it can also be easily implemented using alternative programming language/libraries if needed.

2 Methodology

2.1 Data sets

Details on the sequencing and variant calling (including filtering) for the 16 mosquito samples from Cameroon studied here are given in [5, 10, 15].

As part of the assessment of our method vs. F_{ST}, we simulated a single variant. We used fifty individuals per population for the sweep over unknown genotypes, and for each combination, we converted the appropriate number of samples' genotypes to unknown genotypes before computing the two metrics. For the sweep over population sizes, we increased population sizes in multiples of two.

2.2 Analytical Equations for Probabilities

In diploid organisms, SNPs for individual samples can be thought of as multi-sets over the nucleotides A, T, C, and G. For example, the homozygous A, homozygous T, and heterozygous genotypes would be represented as the following multi-sets, respectively: $\{A, A\}$, $\{T, T\}$, and $\{A, T\}$.

We can calculate the probability that an individual belongs to population one of two conditioned on its genotype as follows:

$$\begin{aligned} P(y=1|gt) &= \frac{P(gt|y=1)P(y=1)}{P(gt)} \\ &= \frac{\frac{N_{gt,1}}{N_1} \cdot \frac{N_1}{N}}{\frac{N_{gt}}{N}} \\ &= \frac{N_{gt,1}}{N_{gt}} \end{aligned} \tag{1}$$

For unknown genotypes, we make the uninformative assumption that the unknown genotype could be any of the possible genotypes with equal probability. In particular, we do not want to assume that we can accurately infer the true genotype of an unknown genotype from the known genotypes among sampled individuals. Additionally, we do not want to infer the class probability based on the distribution of the unknown genotypes across the classes. Note that this is a significant difference between this method traditional human GWAS analysis, because in the latter imputation is often required prior to running LR Tests.

Mathematically, we can define the conditional class probability for the unknown genotype as the union of of the conditional class probabilities for each of the known genotypes. Note that the known genotypes are mutually exclusive.

$$\begin{aligned} P(y=1|gt) &= \frac{P(gt|y=1)P(y=1)}{P(gt)} \\ &= \frac{N_{gt,1} + \frac{1}{3}N_{\{\},1}}{N_{gt} + \frac{1}{3}N_{\{\}}} \end{aligned} \tag{2}$$

$$\begin{aligned} P(y=1|\{\}) &= \frac{P(\{\}|y=1)P(y=1)}{P(\{\})} \\ &= \frac{N_1}{N} \end{aligned} \tag{3}$$

2.3 Logistic Regression Model

Assume that we have N samples with V biallelic positions. Each position has a reference allele and an alternative allele, and at each position, each sample has one of three genotypes (homozygous reference, homozygous alternate, or heterogyzous).

For each position, we encode the variants as a feature matrix $\mathbf{X}$ with dimensions $N \times 3$. We represent each genotype for each position as one of three categorical variables. If sample i has the homozygous reference genotype at position k, then we set $\mathbf{X}_{i,1} = 1$. If sample i has the homozygous alternate genotype at position k, then we set $\mathbf{X}_{i,2} = 1$. If sample i has the heterozygous genotype at position k, then we set $\mathbf{X}_{i,3} = 1$. If the genotype of sample i is unknown at position k, then the row contains zeros in every column.

From the samples' population labels, we define a N-length vector $\mathbf{y}$ of class labels. We then fit the parameters of a Logistic Regression model with the form [7]:

$$P(\mathbf{y}_i = 1|\mathbf{X}_i) = \frac{1}{1 + \exp(-\beta \cdot \mathbf{X}_i + \beta_0)} \tag{4}$$

where $\mathbf{y}_i$ is the class label and $\mathbf{X}_i$ is the feature vector for a single sample i and β is the P-length weight vector

and β_0 is the intercept. We trained the model using Stochastic Gradient Descent (SGD) and an L_2 penalty. (For the experiments in this paper, we performed 10,000 epochs of training for each model.)

In the "standard case", we fit a LogReg model on the feature matrix $\mathbf{X}$ for each position and vector $\mathbf{y}$ of class labels described above.

To adjust the conditional class probabilities, we employ the following revised training procedure. We form a new $3N \times 3$ feature matrix $\tilde{\mathbf{X}}$ and a new $3N$ vector $\tilde{\mathbf{y}}$ of class labels by duplicating each data point three times (since there are three possible genotypes). For unknown genotypes, we set each copy to one of the three known genotypes. Thus, the conditional class probabilities for the known genotypes will incorporate a key assumption of our method: that each unknown genotype has an equal probability of being one of the known genotypes (i.e., "uninformative prior."). We also set the LogReg model intercept to the fraction of samples in class one versus all of the samples and fix the intercept so it is not altered during the SGD training. This ensures that the conditional class probabilities for the unknown genotypes are determined by the ratio of class one samples to all samples. Lastly, we train the weights of the LogReg model using SGD.

Note that for predicting the conditional class probabilities, we utilize the original feature matrix $\mathbf{X}$ and class labels $\mathbf{y}$, regardless of training method.

2.4 Likelihood-Ratio Test

The log likelihood for the Logistic Regression model is given by [7]:

$$\log L(\beta, \beta_0|\mathbf{X}, y) = \prod_{i=1}^{N} \log y_i P(\mathbf{y}_i = 1|\mathbf{X}_i) + (1 - \mathbf{y}_i) \log(1 - P(\mathbf{y}_i = 1|\mathbf{X}_i)) \quad (5)$$

To perform the Likelihood-Ratio Test, two LogReg models are trained. The first model (the alternative), trained as described in Section 2.3, contains additional independent variables (features) not in the null model. (In our case, the null model only contains the intercept and thus, predicts the conditional class probabilities using the ratio of class one samples to all samples.) The weights (β_1, β_0) from the two models are used to compute the log likelihoods. The difference G between the two is calculated by:

$$G = 2(\log L(\beta^1, \beta_0^1|\mathbf{X}^1, \mathbf{y}) - \log L(\beta^0, \beta_0^0|\mathbf{X}^0, \mathbf{y})) \quad (6)$$

The p-value for the difference in log likelihoods is calculated using the χ^2 distribution:

$$p = P[\chi^2(df) > G] \quad (7)$$

where *df* is the difference in the number of degrees of freedom (weights) between the two models.

2.5 Corrected Significance Level

We used a significance level of $\alpha = 0.01$ (1%). Following the method of [6], we performed a PCA analysis of the *Anopheles* SNPs and found that 15 principal components were needed to explain 99.9% of the variance. Using their modified version of Bonferroni correction, we used $0.01/15 = 6.66 \times 10^{-4}$ as the cutoff.

2.6 Ranking SNPs with $\mathbf{F}_{ST}$

To rank the SNPs, we first calculated the the F_{ST} score for each position using VCFTools [3]. Scores which were invalid (nan) or negative were to set to zero. Then, we sorted the SNP positions in descending order by their F_{ST} scores.

2.7 Asaph

Our method was evaluated using Asaph, our toolkit for variant analysis. Asaph was implemented in Python using Numpy / Scipy [18], Matplotlib [9], and Scikit Learn [16] and is available at `https://github.com/rnowling/asaph` under the Apache Public License v2.

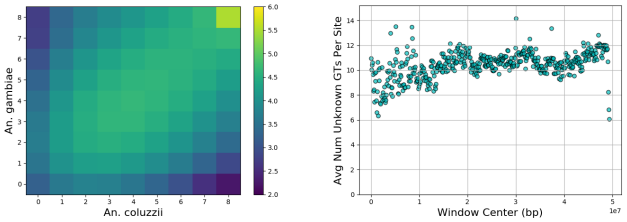
3 Experimental Results

3.1 Genotypes for Many *Anopheles* Variants are Unknown

To motivate our work, we analyzed the prevalence of unknown genotypes among the ≈1.7 million positions described in Section 2.1. For each site, we counted the number of unknown genotypes per species, which is given as a 2D histogram (with log counts) in Figure 1a. The unknown genotypes seemed to occur equally in both species. Fewer than 3% of all positions have known genotypes for each sample, while for as many as 25% of the positions, none of the genotypes are known for any of the samples in at least one population (data not shown).

We also analyzed the presence of unknown genotypes across the 2L chromosome arm. We counted the number of unknown genotypes per site and computed averages over non-overlapping 100 Kbp windows (see Figure 1b. While, the number of unknown genotypes was highest from the beginning of the inversion region (at 25 Mbp) to the end of the arm, on average more than half of the genotypes per site are unknown. Thus, unknown genotypes are highly common for this data set, which makes downstream analysis challenging.

Figure 1: Analysis of Sites on 2L with Unknown Genotypes. (a) Histogram ($\log_{10}$) of Unknown Genotypes Per Site By Species (b) Average Number of Unknown Genotypes Per Site in non-overlapping 100 Kbp Windows



(a) (b)

3.2 Mean Absolute Error of Proposed Training Method

We also evaluated the agreement of the conditional class probabilities computed by Logistic Regression (LogReg) models. For each of 800 SNPs with between zero and all-but-one unknown genotypes sampled from the *Anopheles* data set, we trained models with the standard approach and with our proposed approach described in Section 2.3. We calculated the probability for each of the four possible genotypes using each of the models. Lastly, we calculated the mean absolute error (MAE), broken down by genotype, between the probabilities from the LogReg models and the analytical probabilities.

The MAEs are reported in Table 1. With the standard training method, the LogReg model achieves a MAE as large as 0.23. With the new training approach, the largest MAE is as low as 0.0081. For the case of the unknown genotype, the error is reduced to 0, as expected.

Table 1: Mean Absolute Errors (MAE) of Analytical vs Logistic Regression-Estimated Probabilities

	Standard	**Corrected**
Homo. 1	1.3×10^{-1}	1.5×10^{-4}
Homo. 2	1.3×10^{-2}	8.1×10^{-3}
Het.	1.7×10^{-2}	8.1×10^{-3}
Unknown	2.3×10^{-1}	0.

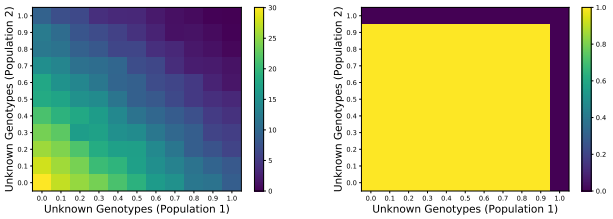
3.3 Varying of the Number of Samples and Unknown Genotypes

The Likelihood-Ratio Test (LR Test) differs from F_{ST} in two significant ways: its p-value incorporates the number of the samples and, because of our proposed training method, the percentage of unknown genotypes is also factored in. We illustrate these differences in comparisons on simulated data (see Section 2.1).

First, we considered a fixed difference where samples in one class have one homogeneous genotype and samples in the second class have the other homogeneous genotype. We swept over different combinations of percentages of samples with unknown genotypes from each population. Except for cases where all of the samples in a single class have unknown genotypes, the F_{ST} scores for all combinations are one. In contrast, the LR Test p-values increase as the percentage of unknown genotypes increase, as desired (see Figure 2).

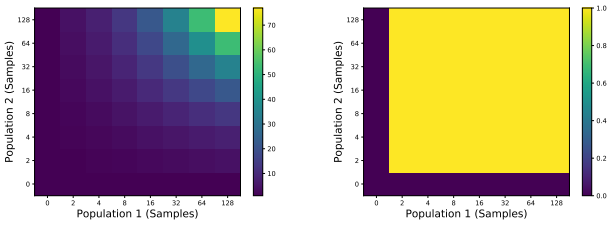
In the second comparison, we re-considered the fixed difference, but with different combinations of sample sizes in each class. We calculated the LR Test p-value and F_{ST} score for each combination (see Figure 3). As before, the F_{ST} scores for each combination were one, except when one of the populations had zero samples. The LR Test p-values decreased as the number of samples increased.

Figure 2: Adjusted Likelihood-Ratio Test p-Values ($-\log_{10}$) and F_{ST} Scores for Different Percentages of Unknown Genotypes for a Fixed Difference



(a) LR Test p-Values (log) (b) F_{ST} Scores

Figure 3: Adjusted Likelihood-Ratio Test p-Values ($-\log_{10}$) and F_{ST} Scores for Different Combinations of Sample Sizes for a Fixed Difference



(a) LR Test p-Values (log) (b) F_{ST} Scores

3.4 Analysis of SNPs from the *Anopheles* Data set

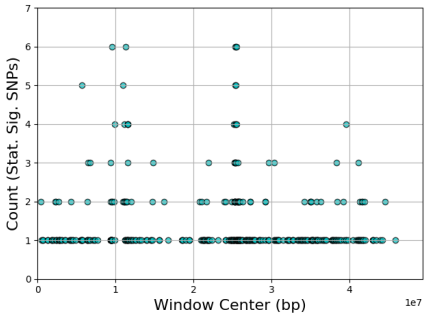
We applied the adjusted Likelihood-Ratio Test (LR Test) to perform single SNP association tests on two data sets of SNPs from the *Anopheles gambiae* and *Anopheles coluzzii* species. We first calculated q-values, a measure of significance in terms of the false discovery rate (FDR) [17, 19]. None of the SNPs, however, satisfied a q-value threshold of 0.01 (FDR of 1%).

Next, we then used the PCA-based method of [6] to determine a less conservative significance threshold (see Section 2.5). Our chosen significance level of $\alpha = 0.01$ (1%) was corrected to $0.01/15 = 6.66 \times 10^{-4}$. At that level, 522 SNPs passed the revised threshold.

For initial validation, we "binned" these 522 SNPs across the 2L chromosome in non-overlapping 10 Kbp windows—combining our method with that of [10]—and found three interesting regions: 10 Mbp, 25 Mbp, and 40 Mbp. Significantly, the 25 Mbp region and 40 Mbp region corresponds to the 2La inversion boundaries, the frequencies of which are known to differ between these samples [10, 15]. The high concentration in the 10 Mbp region is a novel result, and has been provided to our biological collaborators.

We also briefly analyzed the top 20 (as ranked by their p-values) statistically-significant SNPs individually. The first- (position 25,396,564), third- (position 21,707,904), and fifth-ranked (position 25,403,885) SNPs are located within the resistance to dieldrin (Rdl) gene, which has been previously associated with insecticide resistance in *A. gambiae* and other insects [4, 10].

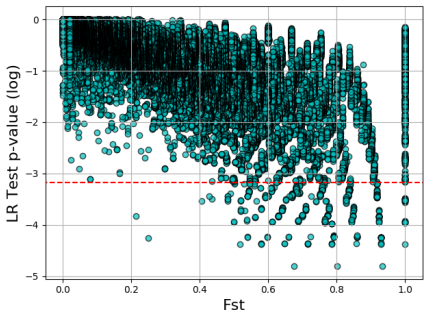
Figure 4: Counts of 522 Statistically-Significant SNPs Appearing in 10 Kbp Windows Across 2L Chromosome



We compared the adjusted LR Test p-values to the F_{ST} scores for the SNPs (see Figure 5). Notably, 1,633 SNPs have F_{ST} scores of 1, but only 20 were found in the set of 522 statistically-significant SNPs. The small number of statistically-significant SNPs with $F_{ST} = 1$ was most likely due to unknown genotypes.

Additionally, the F_{ST} scores of some of the 522 statistically-significant SNPs were as low as 0.2. We attribute this result to our categorical encoding scheme, which considers genotypes, not alleles. In fact, the uncovered 2La inversion breakpoints are only fixed in one species and by definition have non-ideal F_{ST} scores.

Figure 5: Likelihood-Ratio Test p-Values vs F_{ST} Scores. Red dashed line indicates significance threshold.



4 Discussion and Conclusion

The Likelihood-Ratio Test (LR Test) has several properties that make it desirable for population genetics analysis. In particular, unlike the more commonly used F_{ST} metric, the LR Test provides p-value that can be used to identify statistically-significant variants relative to a null model based purely on class probabilities.

Challenges in the sequencing and assembly of insect genomes results in a high propensity for unknown genotypes, as illustrated in Section 3.1. Significantly, we demonstrated in Section 3.3 that our LR Test framework can adjust the calculated p-value in line with the percentage of unknown genotypes and smaller sample sizes to address unknown values without requiring highly difficult and often impossible genotype imputation these species.

Using the adjusted LR Test, 522 *Anopheles* SNPs were found to be statistically significant. Since F_{ST} only uses population frequencies and ignores unknown genotypes in their calculation, only 20 of the 1,633 SNPs with $F_{ST} = 1$ were among the 522 significant SNPs. Significantly, treating the heterozygous genotype separately may help uncover important non-fixed differences such as the ecologically important 2La inversion [10] rediscovered here.

When used in place of F_{ST}, the adjusted LR Test has the potential to substantially reduce false positives without requiring combining multiple loci together, as is often down with window analysis (see [10]). As such, the adjusted LR Test could significantly impact

population genetics by ranking specific sequence-based differences, which will be essential to quickly characterizing and ultimately helping understand speciation in highly similar species.

5 Acknowledgments

The authors would like to thank Nora Besansky, Michael Fontaine, Becca Love, and Aaron Steele for thoughtful discussions that provided the motivation for this effort.

References

[1] Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls. *Nature*, 447(7145):661–678, Jun 2007.

[2] D. J. Balding. A tutorial on statistical methods for population association studies. *Nat Rev Genet*, 7(10):781–791, Oct 2006.

[3] P. Danecek, A. Auton, G. Abecasis, C. A. Albers, E. Banks, M. A. DePristo, R. E. Handsaker, G. Lunter, G. T. Marth, S. T. Sherry, et al. The variant call format and vcftools. *Bioinformatics*, 27(15):2156, 2011.

[4] W. Du, T. Awolola, P. Howell, L. Koekemoer, B. Brooke, M. Benedict, M. Coetzee, and L. Zheng. Independent mutations in the Rdl locus confer dieldrin resistance to *Anopheles gambiae* and *An. arabiensis*. *Insect Mol Biol*, 14(2):179–183, 2005.

[5] M. C. Fontaine, J. B. Pease, A. Steele, R. M. Waterhouse, D. E. Neafsey, I. V. Sharakhov, X. Jiang, A. B. Hall, F. Catteruccia, E. Kakani, et al. Extensive introgression in a malaria vector species complex revealed by phylogenomics. *Science*, 347(6217), 2015.

[6] X. Gao, J. Starmer, and E. R. Martin. A multiple testing correction method for genetic association studies using correlated single nucleotide polymorphisms. *Genetic Epidemiology*, 32(4):361–369, 2008.

[7] D. W. Hosmer Jr., S. Lemeshow, and R. X. Sturdivant. *Applied Logistic Regression*. Wiley, 3 edition, 2013.

[8] B. N. Howie, P. Donnelly, and J. Marchini. A flexible and accurate genotype imputation method for the next generation of genome-wide association studies. *PLOS Genetics*, 5(6):1–15, 06 2009.

[9] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing In Science & Engineering*, 9(3):90–95, 2007.

[10] M. K. N. Lawniczak, S. J. Emrich, A. K. Holloway, A. P. Regier, M. Olson, B. White, S. Redmond, L. Fulton, E. Appelbaum, J. Godfrey, et al. Widespread divergence between incipient *Anopheles gambiae* species revealed by whole genome sequences. *Science*, 330(6003):512–514, 2010.

[11] J. Marchini and B. Howie. Genotype imputation for genome-wide association studies. 11:499 EP –, Jun 2010. Review Article.

[12] J. Marchini, B. Howie, S. Myers, G. McVean, and P. Donnelly. A new multipoint method for genome-wide association studies by imputation of genotypes. *Nat Genet*, 39(7):906–913, Jul 2007.

[13] A. P. Michel, W. M. Guelbeogo, O. Grushko, B. J. Schemerhorn, M. Kern, M. B. Willard, N' F. Sagnon, C. Costantini, and N. J. Besansky. Molecular differentiation between chromosomally defined incipient species of *Anopheles funestus*. *Insect Molecular Biology*, 14(4):375–87, 2005.

[14] A. Miles, N. J Harding, G. Botta, C. Clarkson, T. Antao, K. Kozak, D. Schrider, A. Kern, S. Redmond, I. Sharakhov, et al. Natural diversity of the malaria vector anopheles gambiae. 2016.

[15] D. E. Neafsey, M. K. N. Lawniczak, and D. J. Park. SNP genotyping defines complex gene-flow boundaries among African malaria vector mosquitoes. *Science*, 2984, 2010.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[17] J. D. Storey and R. Tibshirani. Statistical significance for genomewide studies. *Proceedings of the National Academy of Sciences*, 100(16):9440–9445, 2003.

[18] S. v. d. Walt, S. C. Colbert, and G. Varoquaux. The numpy array: A structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22–30, 2011.

[19] J.D. Storey with contributions from A. J. Bass, A. Dabney, and D. Robinson. *qvalue: Q-value estimation for false discovery rate control*, 2017. R package version 2.8.1.

Computing Gene-Disease Associations Efficiently

Kamal Taha
Department of Electrical and Computer Engineering, Khalifa University, UAE
kamal.taha@kustar.ac.ae

Abstract

Most common diseases are complex genetic traits, with genetic contributing to susceptibility to the diseases. Knowing the genes and their variations that involved in a disease is crucial for early intervention and the identification of techniques that can cure the disease. Experimental methods for determining gene–disease associations are laborious and time consuming. This created the need for computational methods to predict the candidate genes associated with diseases, which will be verified using experimental methods. However, most current computational methods may return a large spectrum of candidate genes, which makes their verification by the experimental methods to be time consuming and laborious. We propose in this paper a state-of-the-art biological system called GDL that can overcome the above-mentioned limitation. It does so by short-listing the likely candidate genes involved in a disease to a small and tightly defined group that elicits the disease when work in concert. Since the number of predicted genes is small, the verification of their involvement in the disease by experimental methods will be highly efficient. GDL will help biologists focus their investigation on a small and tightly defined group of genes.

Keywords: Biomedical literature, Gene-disease associations, Genetic illnesses, Information extraction, Text mining.

1 Introduction

Most current computational methods for predicting gene-disease associations may return a *large* spectrum of candidate genes, which makes their verification by the experimental methods time consuming and laborious. Some of these promising methods take advantage of the exponential explosion of biomedical literature. They extract gene-disease associations that appear within the literature. Despite the strength of these methods, however, they suffer the following limitations: (1) they may be suited for *only certain classes* of diseases, and (2) they may predict *a large number* of candidate genes as associated with the disease.

We propose in this paper a state-of-the-art biological system called **GDL** "**G**ene-**D**isease **L**inker" that can overcome the above-mentioned limitations of current computational methods. GDL does so by short-listing the candidate genes to be involved in a disease to a small and tightly defined group that will elicit the disease when work in concert. Since the number of predicted genes is small, the verification of their involvement in the disease by experimental methods will be highly efficient. To ensure that the predicted genes will likely elicit the disease when work in concert, these genes should *possess the characteristics of all the Functional Categories associated with the disease under consideration*. GDL will help biologists focus their gene-disease investigations on small and tightly defined groups of genes. First, GDL extracts the genes that appear within the biomedical literature associated with the Functional Categories involved in the disease. Then, it short-lists these genes by employing the following triple techniques: decision tree [11], logistic regression [3], and chi-squared analysis [9].

First, GDL needs to classify the genes that are likely to be involved in a disease under consideration into the Functional Categories that define the classes of these genes. Towards this, GDL will extract the genes that appear within the biomedical literature associated with each Functional Category involved in the disease. Then, GDL will rank the Functional Categories based on their Information Gains [7]. Subsequently, GDL will construct a decision tree by placing each node representing a Functional Categories in a hierarchical level in the tree that corresponds to its rank. GDL uses logistic regression to estimate the linear decision boundary (i.e., threshold) that divides the class of genes defined by each Functional Category in the decision tree. Finally, GDL will compute the chi-squared value for each path (i.e., branch) in the decision tree to identify the nodes of the path p that yields the highest chi-squared value. Our hypothesis is that the leaf node of the path p contains the smallest short-listed group of genes that are most likely to elicit the disease when work in concert.

The gene-disease associations predicted by GDL can enhance the development of new techniques for preventing, diagnosing, and treating genetic diseases. Below are more specific contributions of the proposed system GDL:

- GDL can help in studying phenotype-genotype relationships, which facilitates genetic testing, monitoring of symptoms, and prognosis. It can be used by medical research centers to develop applications that allow linking existing genetic associations to structured knowledge of phenotypes. This can be done by linking the associations between a group of genes predicted by GDL to structured knowledge of phenotypes. Recall that GDL identifies such a group of genes by locating the leaf node of the path in the decision tree with the highest chi-squared value. The applications can be used for determining which genomic variants affect phenotypes [6, 10]. This will

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

help in explaining diseases, health, and evolutionary fitness.

- After medical research centers verify the gene-disease association predicted by GDL, they can use this information to find new techniques that will improve the quality of life for patients affected by genetic disorders and rare diseases, which increases the life span of the patients.
- It can be used for investigating gene-gene interactions and gene-environment interactions.

To the best of our knowledge, this is the first study that advocate the prediction of the smallest and tightest defined group of genes involves in a disease, where the group has the following properties: (1) it possesses the characteristics of all the Functional Categories associated with the disease, and (2) it is likely to elicit the disease when work in concert. Moreover, this is the first study that investigates the employment of the following triple techniques to short-list the likely genes involved in a disease: decision tree, logistic regression, and chi-squared analysis.

The following is an overview of the sequential processing steps taken by GDL to short-list the likely candidate genes to be involved in a disease *d*:

1) ***Determining the set of genes annotated with all the Functional Categories involved in the disease d*:** For each Functional Category *f* associated with *d*, we select the biomedical literatures associated with *f* from high-quality biological databases (e.g., [17]). GDL will extract the genes that occur within these literatures. It will classify the extracted genes into the Functional Categories that define the classes of genes annotated with their functions. GDL uses Stanford Named Entity Recognizer [12] to tag gene terms in texts. It employs a tokenizer and a stemmer to align the sequence of words in the texts and the names of genes [13].
2) ***Constructing a decision tree:*** GDL will rank the Functional Categories based on their Information Gains [7]. Then, it will construct a decision tree [11] by placing each Functional Category in a hierarchical level in the decision tree that corresponds to its rank.
3) ***Using logistic regression to estimate the linear decision boundary of a Functional Category:*** GDL uses logistic regression [3] to estimate the linear decision boundary (i.e., threshold) that divides the class of genes defined by a Functional Category *f* into two subclasses. It will represent the decision boundary of *f* by a Z-Score [1].
4) ***Using chi-squared analysis to short-list the likely genes to be involved in the disease d:*** Finally, GDL will compute the chi-squared value [9] for each path (i.e., branch) in the decision tree. Our hypothesis is that the path *p* that yields the highest chi-squared value contains the likely genes involved in the disease *d*. Specifically, the leaf node of the path *p* will contain the smallest short-listed group of genes that are most likely to elicit the disease *d* when work in concert

2 Constructing a Decision Tree

GDL employs ID3 algorithm [11] for building decision trees. ID3 measures impurity through entropy. The following are GDL's sequential processing steps for building a decision tree:

1) First, biologists need to construct a table. Each column represents a Functional Category. Each row holds a list of genes annotated with all the Functional Categories involved in the disease.
2) The biologists will need to add a column to the table. In each field of this column, the biologists indicate the *likelihood* that the combination of all the genes in the corresponding row will elicit the disease when work in concert. The biologists can determine this based on the following: (a) the information in the row, and (b) their own domain knowledge and expertise.
3) The table described in steps 1 and 2 will be broken into sub-tables according to the Functional Categories. Each sub-table holds the likelihoods of the genes annotated with only *one Functional Category* to elicit the disease.
4) For each of the sub-tables described in step 3, GDL will calculate the entropy of each *distinct* gene in the sub-table. Entropy (ENT) is computed using Equation 1.

$$ENT = \sum\nolimits_f - p_f \log_2 p_f \qquad (1)$$

where p_f is the probability of each Functional Category *f*. ENT measures the uncertainty. The higher the uncertainty, the higher the entropy is. ENT of a pure table that consists of a single Functional Category is zero, since $p_f = 1$ and $log_2 p_f = 0$ (hence, we achieve minimal impurity). Maximal impurity can be achieved when we have *n* Functional Categories and each happens with equal probability.

5) GDL computes the Information Gain (IG) [2] for each Functional Category *f* based on the entropies of *f* that were calculated in step 4. IG measures the difference in entropy for *f* before and after the data is split on the data of *f* (i.e., before and after the table described in steps 1 and 2 is broken on the data of *f* as described in step 3). It measures the degree of reduction of uncertainty after the data is broken. It is computed as purity degrees of the parent table and weighted summation of impurity degrees of the subset table. IG is computed as follows:

$$IG = ENT(parent) - \text{Weighted Sum of } ENT(Child) \qquad (2)$$

Finally, the Functional Categories are ranked based on their IG values. Then, a decision tree is constructed by placing each node representing a Functional Category *f* in the hierarchical level in the decision tree that corresponds to the rank of *f*. Thus, the root node of the decision tree represents the Functional Category that has largest IG.

Running Example: As a running example throughout the paper, we illustrate how GDL can short-list the likely genes involved in the disease "**deletion syndrome**", using the following five Functional Categories involved in the disease:
- Di-trihydroxycholestanoic acid oxidation/Bile acid (**DAO**).
- Fatty acid oxidation (**FAO**).
- Fatty acid synthesis (**FAS**).
- Peroxisomal membrane proteins (**PMP**).
- Straight chain fatty acids oxidation (**CFAO**).

Example 1: We show how GDL builds the decision tree of the running example based on the likelihood data in Table 1 of genes involvement in the disease. Table 2 shows Table 1 after being broken into 5 sub-tables. Table 3 shows the ENT of each gene. The IG of each Functional Category is calculated based on the data in Table 3, as shown in Table 4. Finally, the Functional Categories are ranked based on their IGs. The decision tree shown in Figure 1 is constructed by placing each node representing a Functional Category in the hierarchical level in the tree that corresponds to its rank.

Table 1: The Likelihood that the combination of the genes in each row elicits the disease "Deletion Syndrome" when work in concert. The set of genes in each row is annotated with all the Functional Categories involved in the disease

PMP	FAS	FAO	DAO	CFAO	Likelihood to Elicit the Disease
g_6	g_{12}	g_{17}	g_{19}	g_{14}	**Yes**
g_6	g_{36}	g_{17}	g_{19}	g_{38}	**No**
g_{35}	g_{53}	g_{25}	g_8	g_{20}	**Yes**
g_{15}	g_{36}	g_{17}	g_{11}	g_{14}	**Yes**
g_{15}	g_{36}	g_{44}	g_{19}	g_{37}	**Yes**
g_6	g_{12}	g_{17}	g_{19}	g_{38}	**Yes**
g_{35}	g_{53}	g_{25}	g_8	g_{14}	**No**
g_{35}	g_{12}	g_{17}	g_{19}	g_{14}	**Yes**
g_{15}	g_{12}	g_{44}	g_8	g_3	**No**
g_{15}	g_{53}	g_{44}	g_{19}	g_{37}	**No**

Table 2: Breaking table 1 into sub-tables according to the Functional Categories. Each sub-table shows the likelihoods of the genes of *one* Functional Category elicits the disease

PMP	L.E.D.	FAS	L.E.D.	FAO	L.E.D.	DAO	L.E.D.	CFAO	L.E.D.
g_6	Yes	g_{12}	Yes	g_{17}	Yes	g_{19}	Yes	g_{14}	Yes
g_6	No	g_{12}	Yes	g_{17}	Yes	g_{19}	No	g_{14}	Yes
g_6	Yes	g_{12}	Yes	g_{17}	Yes	g_{19}	Yes	g_{14}	Yes
g_{35}	Yes	g_{12}	No	g_{17}	Yes	g_{19}	Yes	g_{14}	No
g_{35}	No	g_{36}	No	g_{17}	No	g_{19}	Yes	g_{14}	No
g_{35}	Yes	g_{36}	Yes	g_{25}	Yes	g_{19}	No	g_{17}	No
g_{15}	Yes	g_{36}	Yes	g_{25}	No	g_8	Yes	g_{17}	Yes
g_{15}	Yes	g_{53}	Yes	g_{44}	Yes	g_8	No	g_{25}	Yes
g_{15}	No	g_{53}	No	g_{44}	No	g_8	No	g_{25}	Yes
g_{15}	No	g_{53}	No	g_{44}	No	g_{11}	Yes	g_{25}	No

L.E.D. denotes "Likelihood to Elicit the Disease". The genes are represented numerically for easy reference.

Table 3: ENT of each distinct gene annotated with a Functional Category involved in the disease "Deletion Syndrome". ENT is calculated based on Tables 1 and 2

PMP	ENT	FAS	ENT	FAO	ENT	DAO	ENT	CFAO	ENT
g_6	0.918	g_{12}	0.811	g_{17}	0.722	g_{19}	0.918	g_{14}	0.971
g_{35}	0.918	g_{36}	0.918	g_{25}	1	g_8	0.918	g_{17}	1
g_{15}	1	g_{53}	0.918	g_{44}	0.918	g_{11}	0	g_{25}	0.918

Table 4: The IG of each Functional Category, calculated based on Table 3

Functional Category	PMP	FAS	FAO	DAO	CFAO
IG	0.0202	0.0958	0.1346	0.1448	0.0101

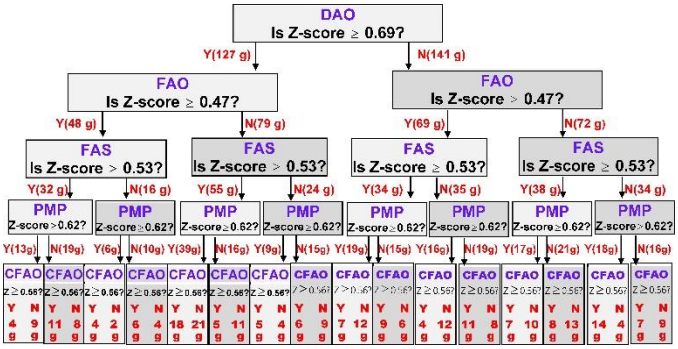


Figure 1: The decision tree of the running example. Y, N, and g denote "Yes", "No", and "gene" respectively. The linear decision boundary for each Functional Category is represented by a Z-score as described in Example 2. The figure shows also the number of genes satisfying the Y and N outcomes of Functional Categories. For example, Y (127 g) denotes that the number of genes satisfying the "Yes" outcome of Functional Category DAO is 127.

3 Using Logistic Regression to Estimate the Linear Decision Boundary of a Functional Category

GDL uses logistic regression to estimate the linear decision boundary of a Functional Category, based on the coefficients B_0 and B_1 [15] for the (multi)linear regression of variables, which are computed as shown in Equations 3 and 4. Equation 5 is used for determining the value of x that makes y zero

$$B_1 = \frac{\sum_{i=1}^{n}(x_i y_i) - \bar{x}\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i^2 - n\,\bar{x}^2} \quad (3)$$

$$B_0 = \bar{y} - B_1\,\bar{x} \quad (4)$$

$$y = B_0 + B_1 x \quad (5)$$

$$\bar{y} = \sum_{i=1}^{n}\frac{y_i}{n}, \quad \bar{x} = \sum_{i=1}^{n}\frac{x_i}{n}, \quad y = \ln\left(odd\left(P(E)\right)\right)$$

$$P(E) = \frac{No.\ of\ \ similar}{No.\ of\ \ similar\ \ +\ No.\ of\ \ non-similar},$$

$$odd\ \ (P(E)) = \frac{P(E)}{1 - P(E)}$$

In the framework of GDL, the decision boundary of a Functional Category *f* is represented by a Z-Score [1]. First, biomedical texts are divided into partitions, where each partition is an incremental percentage of the texts. Let S_i be the set of texts in partition *i*. The estimators B_0 and B_1 of the coefficients for the (multi)linear regression of variables x_1,..., x_n are computed for sets S_1, …, S_n, respectively. In the framework of GDL, x_i denotes the Z-score for set S_i. That is, we use Z-Score to identify the set of genes in set S_i that significantly possesses the characteristics of *f*. This is done by computing the differences between the probabilities of genes occurrence across all function classes. Let: (1) $N_f(S_i)$ be the number of genes that occur in set S_i and are annotated with Functional Category *f*, which is involved in the disease under consideration, (2) $\overline{N}(S_i)$ be the mean of the number of genes that occur in S_i and are annotated with any Functional Category, and (3) σ be the standard deviation of the population. The Z-score of set S_i is computed as Equation 6:

$$Z-score\ (S_i) = \frac{N_f(S_i) - \overline{N}(S_i)}{\sigma} \qquad (6)$$

Example 2: We show how GDL can determine the linear decision boundary for Functional Category DAO (i.e., the Z-score = 0.69 for DAO shown in Figure 1). The decision boundary is determined based on the data in Table 6 (which is in turn calculated from Table 5). Table 5 is constructed as follows. First, the biomedical texts associated with DAO are divided into partitions. Each partition S_i is an incremental percentage of the texts. The following are calculated for each S_i and recorded in the table: (1) $N_{DAO}(S_i)$ (the number of genes that occur in S_i and are annotated with DAO), (2) $N(S_i)$ (the number of genes that occur in S_i and are annotated with *any* Functional Category), and (3) Z-score (S_i) (the Z-score for S_i calculated using Equation 6, where $\overline{N}(S_i)$ is the *mean* of the number of genes that occur in S_i and are annotated with any Functional Category). The coefficients B_0 and B_1 are computed using Equations 3 and 4 based on the intermediate data in Table 6 (which is in turn calculated from Table 5).

Table 5: The Z-score of each partition S_i of texts selected in an incremental percentage of all the texts associated with Functional Category DAO, which is involved in the disease of the running example.

Z-score (S_i)	$N_{DAO}(S_i)$	$N(S_i)$	**Percentage of selected texts**
0.107	17	47	20%
0.872	38	83	40%
-0.625	41	137	60%
-1.08	48	184	80%
-2.64	62	268	100%

Table 6: Intermediate data for calculating B_0 and B_1. The data is computed using equations 3 and 4 based on Table 5

Z-score	P(E)	odd(P(E))	ln(odd(P(E)))
0.107	0.266	0.362	-1.016
0.872	0.314	0.458	-0.781
-0.625	0.230	0.299	-1.207
-1.08	0.207	0.261	-1.343
-2.64	0.188	0.232	-1.461

x and *y* in equations 3-5 denote *Z-score* and *ln(odd(P(E)))*, respectively in the table

4 Using Chi-Squared Analysis to Short-List the Genes that are Likely to be Involved in a Disease

A chi-squared value (χ^2) is computed as follows: $\chi^2 = (O-E)^2/E$, where *O* and *E* are the observed and expected frequencies of the data, respectively. The expected frequency (*E*) for each possible value of the variable is computed as follows: $E = np$, where *n* is the size of the sample and *p* is the relative frequency (or probability).

Example 3: Using the decision tree in Figure 1, we show how GDL uses chi-squared analysis to identify the leaf node containing the likely short-listed genes involved in the disease "deletion syndrome". As Figure 2 shows, there are 16 paths in the tree. GDL computed the chi-squared value for each path. Path 6 achieved the highest chi-squared value. Thus, the leaf node of path 6 contains the short-listed genes. The 5 genes satisfying outcome Y of Functional Category CFAO within the leaf node of path 6 are the short-listed genes that are: (1) likely to be involved in the disease, and (2) annotated with all Functional Categories involved in the disease.

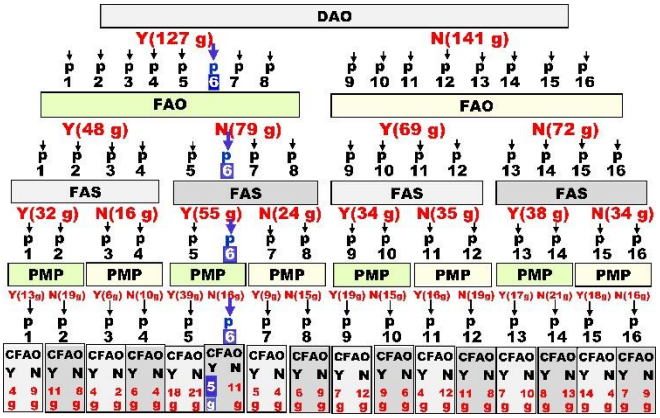


Figure 2: The 16 paths in the decision tree of the running example. Each path is denoted by a combination of: an arrow ↓, a letter "p" underneath the arrow, and a digit underneath the letter. The digit represents the path's number. For example, $\overset{p}{3}$ denotes path number 3. GDL identified the 5 genes that satisfied outcome Y of Functional Category CFAO within the leaf node of path 6 as the short-listed genes that are: (1) likely to be involved in the disease, and (2) annotated with all the Functional Categories involved in the disease "deletion syndrome". Path 6 is marked with blue background.

5 Experimental Results

We implemented GDL in Java, run on Intel(R) Core(TM) i7 processor, with a CPU of 2.70 GHz and 16 GB of RAM, under Windows 10. We experimentally evaluated the quality of GDL for predicting and short-listing the likely genes involved in the disease **"Acyl-CoA Oxidase Deficiency"**. We evaluated the quality of GDL by comparing the list of genes it predicted as the likely to be involved in the disease with the actual list of genes already known to be involved in the disease. We considered the actual list of genes known to be involved in the disease as ground-truth data. Table 7 shows the list of Functional Categories involved in the disease. We retrieved 217 PubMed texts associated with the five Functional Categories according to their entries in the UniProtKB database [17]. Let f be one of the five Functional Categories in Table 7. GDL extracted the list L_f of genes that occur within the texts associated with f among the 217 PubMed texts. GDL stored this information in a genes table with an entry for each of the five Functional Categories.

Table 7: List of Functional Categories involved in the disease "Acyl-CoA Oxidase Deficiency" and their abbreviations

Functional Category	Abbreviation
Di-trihydroxycholestanoic acid beta-oxidation	**DABO**
Long-chain dicarboxylic acids oxidation	**LDAO**
Fatty acid synthesis/PUFAS synthesis	**FAPS**
Straight chain fatty acids beta-oxidation	**SFABO**
Branched-chain fatty-acid oxidation	**BFAO**

GDL ranked the paths in the decision tree based on their chi-squared values. As Figure 3 shows, path 15 achieved the highest chi-squared value. GDL identified the 5 genes satisfied outcome Y of Functional Category LDAO within the leaf node of path 15 as the short-listed group of genes that are likely to elicit the disease "Acyl-CoA oxidase deficiency" when work in concert.

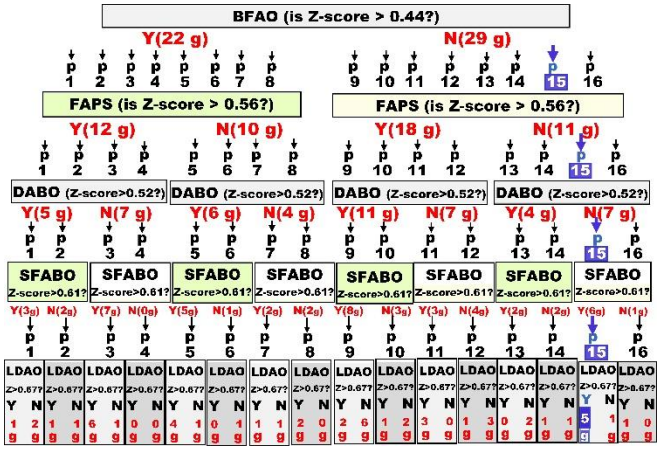


Figure 3: The paths in the decision tree of the disease "Acyl-CoA oxidase deficiency". GDL identified the 5 genes satisfied outcome Y of LDAO within the leaf node of path 15 as the most likely to be involved in the disease.

For each of the top-3 paths ranked by GDL, we measured the Recall, Precision, and F-value of the predicted genes contained in the leaf node of the path. We used the following metrics: Recall= C_g/N_g, Precision = C_g/M_g, and F-value = (2 Recall * Precision)/(Recall+ Precision), where C_g is the number of *correctly* predicted genes in the path's leaf node, N_g is the number of actual genes involved in the disease, and M_g is the number of predicted genes in the lead node. As Figure 4 shows, the *F-value* of a path increases as its rank increases (i.e., the top-ranked path achieved the highest F-value). This is an evident of the accurate identification, classification, and ranking of paths.

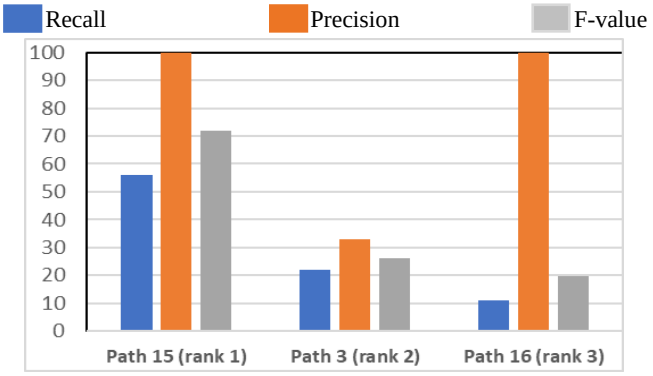


Figure 4: The Recall, Precision, and F-value for predicting the genes contained in the leaf node of each of the top-3 ranked paths in the decision tree.

We also evaluated the accuracy of GDL in terms of the distances between the positions of the genes in the list ranked by GDL and the positions of the same genes in lists ranked by the standard network metrics (i.e., Degree, Closeness, and Betweenness metrics [16]), using the disease-specific gene interaction network. Specifically, we measured the distances between: (1) the positions in the list ranked by GDL that belong to the genes contained in the leaf nodes of the *top-3 ranked paths*, and (2) the positions of the same genes in the lists ranked by the standard network metrics using the disease-specific gene interaction network. Intuitively, the smaller the distances the better GDL. The following is the procedure we considered for ranking the list of genes predicted by GDL:

- Each of the 5 genes in path 15 *(which is ranked first)* is considered to be in **position # 1** in the list.
- Each of the 6 genes in path 3 *(which is ranked second)* is considered to be in **position # 6** in the list.
- The gene in path 16 *(which is ranked third)* is considered to be in **position # 12** in the list.

A ranking is a permutation of the integers 1, 2, ... For each of the top-3 ranked paths, we measured the *average Euclidean Distance* between the positions in the list ranked by GDL that belong to the genes contained in the leaf node of the path, and the positions of the same genes in the lists ranked by the standard network metrics. We used the Euclidean Distance measure shown in Equation 7. Figure 5 shows the results.

$$d(p, m) = \frac{\sum_{g \in p} \left| \sigma_m(g) - \sigma_p(g) \right|}{|p|} \quad (7)$$

where:

- $d(p, m)$: The *average Euclidean Distance* between the positions of path p and network metric m.
- $\sigma_p(g)$: The position in the list ranked by GDL that belongs to gene g, which is contained in the leaf node of path p.
- $\sigma_m(g)$: The position of gene g in the list ranked by network metric m.

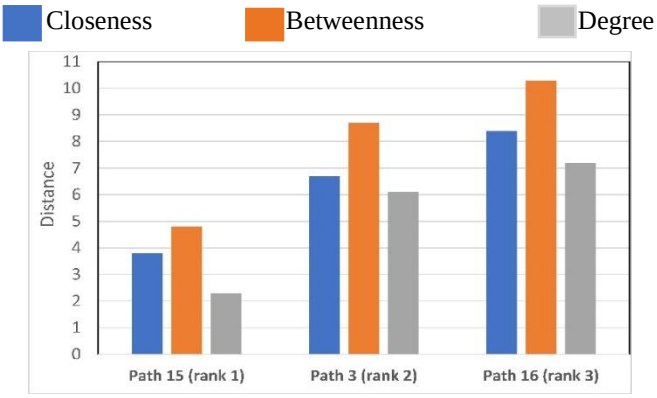


Figure 5: The *average* Euclidean Distance between the positions of the genes contained in each of the top-3 ranked paths and the positions of the same genes in each of the lists ranked by standard network metrics. Intuitively, the smaller the average distance the more accurate the path.

As Figure 5 shows, the *Euclidean Distances* between the positions of the genes in the list ranked by GDL and the positions of the same genes in the lists ranked by the standard network metrics are small. Moreover, these distances tend to decrease as the rank of a path increases, where the top-ranked path (i.e., path 15) achieved the smallest distances.

6 Conclusion

We proposed in this paper a state-of-the-art biological system called GDL that can help biologists short-list the candidate genes that are likely to elicit a disease when work in concert. This helps biologists focus their investigation of the likely genes involved in a disease on a *small and tightly defined* group of genes. This overcomes the limitations of most current approaches that predict a large spectrum of candidate genes, which makes the verification of these genes by experimental methods time consuming and laborious. Moreover, these approaches may be suited for *only certain classes* of diseases. First, GDL constructs a decision tree by ranking the Functional Categories involved in the disease. After employing logistic regression to estimate the linear decision boundaries of the Functional Categories, GDL uses chi-squared analysis to identify the path p that yields the highest chi-squared value. Our hypothesis is that the leaf node of p contains the smallest and tightest defined group of genes involves in a disease, where the group has the following properties: (1) it possesses the characteristics of all the Functional Categories involved in the disease, and (2) it is likely to elicit the disease when work in concert. We experimentally evaluated the quality of GDL for predicting the likely genes involved in the disease "Acyl-CoA Oxidase Deficiency". Results showed high prediction accuracy.

References

[1] A. Wong, H. Shatkay. Protein Function Prediction using Text-based Features extracted from the Biomedical Literature: The CAFA Challenge. BMC Bioinformatics 2013, 14(Suppl 3):S14.

[2] Bauer S, et al. (2010) GOing Bayesian: model-based gene set analysis of genome-scale data. Nucleic Acids Res, 38:3523.

[3] David Hosmer (2013). Applied logistic regression. Hoboken, New Jersey: Wiley. ISBN 978-0470582473.

[4] Hettne et al., "The implicitome: A resource for rationalizing gene-disease associations", 2016, PloS one 11 (2).

[5] H. Zhou, J. Skolnick, "A knowledge-based approach for predicting gene-disease associations", *2016*, Bioinformatics, 32(18).

[6] Morbid Map of the OMIM downloads (2017). Available at: http://www.omim.org/downloads.

[7] O. Al-Jarrah, O Alhussein, P Yoo, S Muhaidat, K Taha, and K. Kim. "Data Randomization and Cluster-Based Partitioning for Botnet Intrusion Detection". *IEEE Transactions on Cybernetics*. 2015, 46(8): 1796-1806.

[8] Ozgür A, Vu T, Erkan G, Radev DR. Identifying gene-disease associations using centrality on a literature mined gene-interaction network. *Bioinformatics* 2008, 24(13), i277 – i285.

[9] P.E. Greenwood, M.S. Nikulin. (1996) *A guide to chi-squared testing*. Wiley, New York.

[10] R. Al-Dalky, Taha, K., Dirar Al Homouz. "Applying Monte Carlo Simulation to Biomedical Literature to Approximate Genetic Network". *IEEE/ACM Transactions on Computational Biology and Bioinformatics*. 2016, 13(3), pp. 494 - 504.

[11] Sung-Hyuk Cha; Charles C Tappert (2009). "A Genetic Algorithm for Constructing Compact Binary Decision Trees". *Journal of Pattern Recognition Research. 4 (1): 1–13.*

[12] Stanford Tokenizer, POS Tagger, and Named Entity Recognizer (2017): http://nlp.stanford.edu/software/

[13] Taha, K. "Extracting Various Classes of Data from Biological Text using the Concept of Existence Dependency". *IEEE Journal of Biomedical and Health Informatics (IEEE J-BHI)*, 2015, Vol. 19, issue 6, pp. 1918 - 1928.

[14] Taha, K. "Determining the Semantic Similarities among Gene Ontology Terms". *IEEE Journal of Biomedical and Health Informatics*, 2013, 17(3).

[15] Tue Tjur (2009). "Coefficients of determination in logistic regression models". American Statistician: 366–372.

[16] Taha, K., Yoo, P. "Using the Spanning Tree of Mobile Communication Network for Identifying Criminal Leaders". *IEEE Transactions on Information Forensics & Security*, 2016, Vol. 12, issue 2, pp. 445 – 453.

[17] UniProt: The UniProt Consortium. Activities at the Universal Protein Resource. Nucleic Acids Research, 2014, Vol. 42, D191–D198.

Searching Jointly Correlated Gene Combinations

Yuanfang Ren*, Ahmet Ay◇, Travis A. Gerke† and Tamer Kahveci*

* Department of Computer and Information Science and Engineering, University of Florida
Gainesville, FL, 32611, USA
(yuanfang, tamer)@cise.ufl.edu

◇ Departments of Biology and Mathematics, Colgate University
Hamilton, NY, 13346, USA
aay@colgate.edu

† Moffitt Cancer Center
Tampa, FL, 33607, USA
travis.gerke@moffitt.org

Abstract

Gene expression associations play an essential role to decipher functions of genes and their interactions. Correlation score between pairs of genes is usually utilized to associate two genes. However, the relationship between genes is often more complex; multiple genes might collaborate to control the transcription of a gene. In this paper, we introduce the problem of searching pairs of genes, which collectively correlate with another gene. This problem is computationally much harder than the classical problem of identifying pairwise gene associations. Exhaustive search is infeasible for the entire human transcriptome also; since for m genes, there are $O(m^3)$ possible gene combinations. Our method builds three filters to avoid computing the association for a large fraction of gene combinations, which do not produce high correlation. Our experiments on a prostate cancer dataset demonstrate that our method produces accurate results at the transcriptome level in practical time. Moreover, our method identifies biologically novel results which classical pairwise gene association studies are unlikely to discover.

Keywords: Pearson's correlation, pairwise correlation, joint correlation, prostate cancer.

1 Introduction

Gene expression analysis is key for understanding the mechanisms of central biological processes, which often small changes in gene transcription levels may have profound effects such as cancer. Analysis of gene transcription patterns has been shown to be helpful in numerous applications such as understanding cellular functions of genes [2], progression of major disorders [16] and cellular responses to external stimulants [8]. One of the first steps in studying gene expression is to explore the associations between genes. Two standard ways to formulate such associations mathematically are to compute the correlations between gene transcription levels using Pearson correlation [15, 18] or Spearman correlation [17, 4, 9]. In this paper, we focus on Pearson's correlation function as it is arguably the most commonly used correlation function, and as it takes the gene transcription levels into consideration.

Although correlation studies on transcription datasets are frequently done, many exhibit a major limitation; the correlations are often computed only for pairs of genes. This is a natural outcome of the correlation functions, including Pearson's correlation, as they work only for two genes at a time. The relationships between genes however are more complex as sets of genes collaboratively can affect the transcription of other genes. Particularly in a cohesive expression network, the identification of subtly interacting gene sets that comprise networks remains to be a challenging task. In the literature, a subset of such collaborative relationships are defined as SSL (Synthetic Sickness and Lethality) interactions [23, 13, 7]. As such, many existing gene expression networks in popular usage have been defined on the basis of biological inference, not through computational search [19]. SSL interactions describe how groups of genes affect an outcome, such as survivability and mobility of cells, instead of the transcription of other genes. As we explain later in detail, here we consider a mathematically different

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

problem, and aim to find groups of genes which collectively correlate with a target gene rather than an outcome in phenotype. Exhaustively searching for groups of genes which collectively regulate or are associated with others is a computationally challenging task. Some attention has previously been given to gene expression pairs that jointly correlate with a binary phenotype [6]. However, to the best of our knowledge, no algorithms have been proposed to efficiently seek gene expression pairs that jointly correlate with a third transcript. In the following, we briefly present a conceptual description of the problem considered in this paper. We present a mathematically precise definition in Section 2.1 after defining proper terms and variables.

Problem definition. Consider a dataset of n samples with each sample having the transcription levels of a set of m genes. Consider three genes g_i, g_j, and g_k from this set. We say that g_i and g_j *jointly correlate* with g_k if the following three conditions hold:

(i) g_i and g_k do not correlate.

(ii) g_j and g_k do not correlate.

(iii) Cumulative transcription levels of g_i and g_j correlate with g_k.

Figure 1 explains this problem on a real example. In this example, we focus on three genes: CLOCK, MED4 and RB1. We obtained gene expression levels from 333 prostate cancer patients from the The Cancer Genome Atlas (TCGA) [1] database (See Section 3 for details on this dataset). We present the scatter plots of the transcription levels for different combinations and compute their correlations. We observe that CLOCK and MED4 genes have low correlation with RB1. However, the sum of the transcription levels of CLOCK and MED4 genes, highly correlate with RB1. Thus, genes CLOCK and MED4 jointly correlate with gene RB1. In this paper, we would like to find all such jointly correlated genes.

Joint correlations between genes may reveal subtle, yet important relationships, which govern transcription of genes. Because the relevant gene triads cannot be identified through classical pairwise gene correlation analyses, this is a computationally challenging task. The challenge arises due to the geometric increase in the number of combinations of genes. More specifically, there are $m \times \binom{m-1}{2}$(i.e. $O(m^3)$) gene combinations. For instance, the human transcriptome has around 20,000 well-annotated genes, which translates into about 4×10^{12} possible combinations, making it infeasible to use an exhaustive approach.

Our Contributions. In this paper, we address the problem of finding jointly correlated gene combinations. We develop three efficient filters to avoid an exhaustive search. We perform extensive experiments on a real gene expression dataset obtained from TCGA database. Our results demonstrate that our method has very high accuracy, and it is dramatically faster than the classical exhaustive approach. Further analysis of the results obtained by our method reveals that there are many substantial joint correlations between genes that are impossible to determine using existing strategies.

We outline the rest of the paper as follows. We formally define the problem and present our method in Section 2. We discuss our experimental results in Section 3 and conclude in Section 4.

2 Methods

We developed three efficient filters to dramatically reduce the time to find joint correlations. Section 2.1 provides the key terms used in our method. Section 2.2 describes our three filters.

2.1 Preliminary terms

Here, we define the key terms which are needed to describe our method. We start by taking a look at the calculation of the Pearson's correlation between two genes.

Assume that we are given a dataset containing the transcription levels of m genes for n samples. We denote the transcription of the ith gene, g_i, for all the n samples with vector $X_i = [x_{i1}, x_{i2}, \ldots, x_{in}]$, where x_{ij} is the transcription level of g_i for the jth sample. Let us denote the standard deviation of vector X_i with $\sigma(X_i)$, and the covariance between vectors X_i and X_j with $cov(X_i, X_j)$ respectively. We denote the Pearson's correlation function between genes g_i and g_j with $f(g_i, g_j)$ and calculate it as:

$$f(g_i, g_j) = \frac{cov(X_i, X_j)}{\sigma(X_i)\sigma(X_j)}.$$

Next, we define a key term which is needed to formulate the problem considered in this paper.

Definition 1. (JOINT CORRELATION). *Given three genes g_i, g_j and g_k, two real valued parameters $\alpha > 0$ and $\beta > 0$, and the correlation thresholds $\epsilon, \epsilon' \in [0, 1]$ ($\epsilon' \leqslant \epsilon$), we define the joint correlation between the pair (g_i, g_j) and the target gene g_k as the Pearson's correlation between vectors $\alpha X_i + \beta X_j$ and X_k, and denote it with $f(\alpha g_i \oplus \beta g_j, g_k)$. We say that genes g_i and g_j* jointly correlate *with g_k if all of the following three conditions hold:*

(1) $|f(\alpha g_i \oplus \beta g_j, g_k)| \geqslant \epsilon$,

(2) $|f(g_i, g_k)| < \epsilon'$,

(3) $|f(g_j, g_k)| < \epsilon'$.

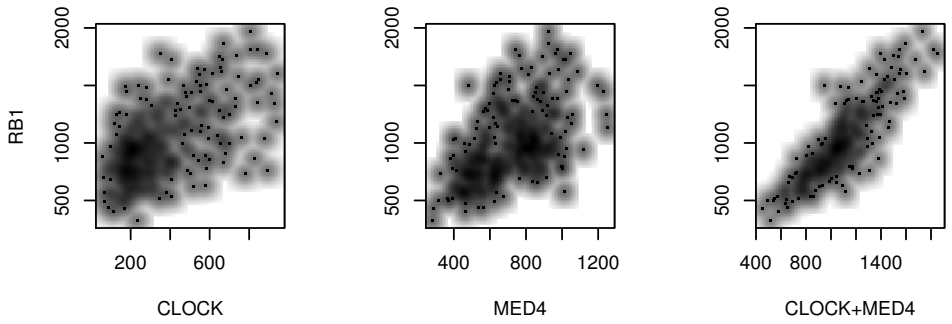


Figure 1: An example of jointly correlated gene combination. Left: scatter plot of the expression levels of genes CLOCK and RB1 over 333 samples (Pearson's correlation = 0.569). Middle: same scatter plot for genes MED4 and RB1 (Pearson's correlation = 0.527). Right: scatter plot of the cumulative transcription of CLOCK and MED4 against RB1 (Pearson's correlation = 0.864).

An important observation following from Definition 1 is that scaling the coefficients of gene combinations by a constant κ does not change their joint correlation. In other words,

$$f(\alpha g_i \oplus \beta g_j, g_k) = f(\kappa\alpha g_i \oplus \kappa\beta g_j, g_k).$$

Thus, using $\kappa = 1/(\alpha + \beta)$, we get

$$f(\alpha g_i \oplus \beta g_j, g_k) = f(g'_i \oplus g'_j, g_k),$$

where $g'_i = \frac{\alpha}{\alpha+\beta} g_i$ and $g'_j = \frac{\beta}{\alpha+\beta} g_j$.

The above equation implies that after preprocessing (i.e., multiplying the transcription value of genes g_i and g_j with $\frac{\alpha}{\alpha+\beta}$ and $\frac{\beta}{\alpha+\beta}$, respectively), we simplify the calculation of the joint correlation by getting rid of the parameters α and β. To simplify our notation, in the rest of this paper, unless otherwise specified, we assume that the transcription values of all genes are already multiplied with these constants, and thus $\alpha = \beta = 1$.

Next, we derive the formulation for the joint correlation of three genes:

$$\begin{aligned} f(g_i \oplus g_j, g_k) =& \frac{cov(X_i + X_j, X_k)}{\sigma(X_i + X_j)\sigma(X_k)} \\ =& \frac{cov(X_i, X_k) + cov(X_j, X_k)}{\sigma(X_i + X_j)\sigma(X_k)} \\ =& \frac{\sigma(X_i)}{\sigma(X_i + X_j)} \frac{cov(X_i, X_k)}{\sigma(X_i)\sigma(X_k)} + \frac{\sigma(X_j)}{\sigma(X_i + X_j)} \frac{cov(X_j, X_k)}{\sigma(X_j)\sigma(X_k)} \\ =& \frac{\sigma(X_i) f(g_i, g_k) + \sigma(X_j) f(g_j, g_k)}{\sigma(X_i + X_j)}. \end{aligned}$$

From the above formulation, we obtain the relationship between the joint correlation and pairwise Pearson's correlation.

Notice that finding the jointly correlated sets of genes can be solved by exhaustively trying all combinations of three genes g_i, g_j and g_k. Although, this would yield correct results, it is computationally prohibitive as the number of combinations is massive when scaled to the entire transcriptome for many organisms. Next, we describe how we solve this problem efficiently. Our method avoids exhaustively testing the geometrically growing search space by developing three filters to quickly remove a substantial fraction of gene combinations, which are guaranteed to not jointly correlate. We discuss these three filters next.

2.2 Filters

In this section, we describe our three filters in detail. Among these, the first two are *unconditional*; the decision to filter g_i does not depend on g_j. The third filter is *conditional* as it considers the g_j level while making a decision on g_i. All these filters utilize the relationship between the joint correlation of three genes and pairwise Pearson's correlation (see section 2.1 for the derivation).

$$f(g_i \oplus g_j, g_k) = \frac{\sigma(X_i) f(g_i, g_k) + \sigma(X_j) f(g_j, g_k)}{\sigma(X_i + X_j)} \quad (1)$$

Filter 1. We obtain our first filter directly from the problem definition. We only consider g_i for joint correlation with g_k if

$$|f(g_i, g_k)| < \epsilon'$$

as neither gene in the gene combination correlates with the target gene individually if they jointly correlate with the target gene.

This filter gives the filter condition for every gene without considering their relationship with others. Thus, it has small computational cost. However, the bound for this filter is loose, and thus we expect to have many gene combinations to remain for the next two filters.

Filter 2. The second filter takes gene g_j into consideration, which leads to a tighter bound. To do this while ensuring that the filter is unconditional, it considers the maximum possible absolute correlation value between g_j and g_k as well as the minimum standard deviation between g_i and g_j. It works as follows.

For each target gene g_k, consider the genes in the group which have positive correlation values with g_k. For gene g_i, we only consider the genes whose standard deviations are smaller than that of g_i. It is worth mentioning that no combination of g_i and g_j is missed in this way. Given $\sigma(X_j) \leqslant \sigma(X_i)$, we have the following formulation:

$$\begin{aligned} f(g_i \oplus g_j, g_k) =& \frac{\sigma(X_i) f(g_i, g_k) + \sigma(X_j) f(g_j, g_k)}{\sigma(X_i + X_j)} \\ \leqslant& \frac{\sigma(X_i)[f(g_i, g_k) + f(g_j, g_k)]}{\sigma(X_i + X_j)}. \end{aligned} \quad (2)$$

Then, we obtain our second filter by replacing two terms in this inequality. We explain how we do this next. Let us denote the set of genes with standard deviation less than or equal to that of gene g_i with

$$A_i = \{g_t |\ \sigma(X_t) \leqslant \sigma(X_i)\}.$$

Also, let us denote the set of genes which are in the same group as g_i with respect to the target gene g_k with

$$B_{i,k} = \{g_t |\ f(g_t, g_k) \cdot f(g_i, g_k) \geqslant 0\}.$$

Consider

$$g_r = \text{argmax}_t \{f(g_t, g_k) |\ g_t \in A_i \cap B_{i,k}\}.$$

We have $f(g_r, g_k) \geqslant f(g_j, g_k)$ since $g_j \in A_i \cap B_{i,k}$. Now, consider

$$g_s = \text{argmin}_t \{\sigma(X_i + X_t) |\ g_t \in A_i\}.$$

Similarly, we have $\sigma(X_i + X_s) \leqslant \sigma(X_i + X_j)$ as $g_j \in A_i$. Using these inequalities, we replace the terms $f(g_j, g_k)$ and $\sigma(X_i + X_j)$ in Equation 2 with $f(g_r, g_k)$ and $\sigma(X_i + X_s)$, respectively. Thus, our second filter removes g_i from consideration if

$$f(g_i \oplus g_j, g_k) \leqslant \frac{\sigma(X_i)[f(g_i, g_k) + f(g_r, g_k)]}{\sigma(X_i + X_s)} < \epsilon.$$

Notice that, the transformation above eliminates the term g_j from the inequality. This transformation seems more complicated than the original one in Equation 1, however, it allows for a much more efficient implementation.

Filter 3. We expect our second filter to yield false positives when the standard deviation of g_i, $\sigma(X_i)$, becomes too large. Our third filter deals with such false positives by actually taking the transcription values of g_j into account. Instead of introducing the two terms g_r and g_s into Equation 2 as we did in Filter 2, we only replace the term $f(g_j, g_k)$ with $f(g_r, g_k)$. Thus, our third filter removes the combination g_i and g_j if

$$f(g_i \oplus g_j, g_k) \leqslant \frac{\sigma(X_i)[f(g_i, g_k) + f(g_r, g_k)]}{\sigma(X_i + X_j)} < \epsilon.$$

That is, g_i and g_j cannot jointly correlate with g_k if

$$\sigma(X_i + X_j) > \frac{\sigma(X_i)[f(g_i, g_k) + f(g_r, g_k)]}{\epsilon}.$$

An interesting question at this point would be: why do we need to use Filter 3 if we need to use the transcription of all the three genes g_i, g_j and g_k? In other words, why do we not simply compute the joint correlation $f(g_i \oplus g_j, g_k)$ directly and resort to a filter? The answer lies in the final inequality of Filter 3 above. It gives us a lower bound for the standard deviation if g_i and g_j do not jointly correlate with g_k. As we precompute the standard deviation for all gene pairs, we utilize this information to avoid computing the joint correlations of all candidate combinations. Recall that in total there are only $\binom{m}{2}$ pairs as compared to $m \times \binom{m-1}{2}$ three gene combinations. Thus, this filter has the potential to eliminate many false joint correlations at a little expense which is already computed as a preprocessing step to our algorithm.

Now that we have demonstrated our method; we next describe an application of this method to a real gene expression dataset.

3 Results

In this section, we experimentally test our method on a real dataset, and measure its performance in terms of accuracy and running time. Moreover, we also discuss the biological characteristics of jointly correlated gene combinations we have found. Notice that, in all experiments, we set the parameter α and β to 1. In the following, we describe the dataset used in our experiments.

Real dataset. We use the processed Prostate Adenocarcinoma (PRAD) RNA-seq data [12] downloaded from TCGA. This dataset consists of transcription levels of 20,531 genes for 333 samples. Notice that, all transcription levels of genes have been normalized using log transformation. We remove genes with consistently low expression throughout a large majority of the samples in the following way. If the transcription level of a gene is less than 10 for more than 95 percent of samples, we say that this gene is not expressed at a sufficient level, and remove it. After the preprocessing step, a total of 16,513 genes remain in our dataset.

3.1 Evaluations on real datasets

In this section, we measure the performance of our method on the real dataset described above in terms of its running time and accuracy.

3.1.1 Evaluation of running time

In the real dataset, three parameters affect the running time of our method: the number of genes, and the

two correlation thresholds ϵ and ϵ'. Out of these three parameters, only the first and the third affect the running time of the exhaustive search. We experimentally evaluate the impact of all of the these parameters on the performance of our method next.

Impact of number of genes. As the exhaustive approach does not scale to the entire dataset, we run experiments on small subsets of our dataset with varying number of genes. In particular, we vary the dataset size from 200 to 2000 at increments of 200. For each dataset size, we construct 10 different gene sets by randomly selecting a subset of genes and report the average and the standard deviation of the running time for each dataset size. We fix the thresholds ϵ and ϵ' to 0.75 and 0.6 respectively. Figure 2a reports the results for our method and exhaustive search. When the number of genes reaches 800, exhaustive search takes excessively long time (i.e., more than 10 hours). Thus, we do not report the running time of exhaustive search when number of genes exceeds 800. We observe a huge difference in the running time of the two methods. Our method runs several orders of magnitude faster than the exhaustive search. As the number of genes increases, the gap between the running time of exhaustive approach and our method grows much faster. It is worth noting that the standard deviation of the running time in this experiment is very low (less than 0.005 times the mean) for both exhaustive search and our method for all dataset sizes, as such the error bars indicating the standard deviation in Figure 2a are almost overlapping. This is very promising, as it suggests that our results are stable, and thus the gap between the running time of our method and that of exhaustive search does not change by altering the gene set.

In summary, we observe that the exhaustive search is not practical for large number of genes, whereas our method scales to large datasets.

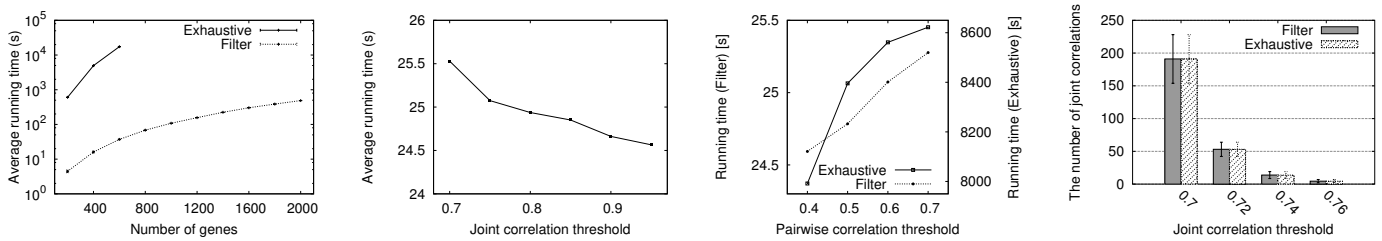
Impact of joint correlation threshold. Next, we turn our attention to the effect of the threshold ϵ. We vary the threshold ϵ from 0.7 to 0.95, and fix the size of gene set and ϵ' to 500 and 0.6, respectively. We repeat the experiment on 10 different datasets and report the average running time. Figure 2b plots the results for our method. We do not plot the running time of the exhaustive method as the parameter ϵ has no influence on its performance. It takes about 2 hours 40 minutes for exhaustive search to run on a dataset of this size. Similar to the previous experiment, our method runs several orders of magnitude faster than the exhaustive method for all values of ϵ. Moreover, we observe that the threshold ϵ has slight effect on the running time of our methods. As the value of ϵ increases, the running time of our method slightly reduces.

Impact of pairwise correlation threshold. Finally, we explore the effect of the threshold ϵ'. We vary the value of threshold ϵ' from 0.4 to 0.7 at increments of 0.1. We set the number of genes and ϵ to 500 and 0.75, respectively. We repeat the experiment on 10 different datasets and report the average running time. Figure 2c shows the results for both methods. Similar to the previous experiments, our method runs several orders of magnitude faster than the exhaustive method for all values of ϵ'. Moreover, we also observe that when ϵ' grows, the running time slightly goes up. This is expected as smaller ϵ' value makes Filter 1 remove more g_i. However, the increase of the running time is not notable. This is because the effect of the threshold ϵ' mainly depends on the distribution of pairwise correlation values. In other words, a large number of pairwise correlation values on our dataset are smaller than 0.4.

In summary, our experiments demonstrate that both methods depend mainly on the number of genes. As the number of genes grows, the running time of both methods increases. However, our method runs several orders of magnitude faster than the exhaustive search. The joint correlation threshold does not affect the running time of exhaustive search, but has negligible impact on our method. Moreover, the pairwise correlation threshold does have some effect on the running time of both methods. However its effect mainly depends on the distribution of pairwise correlation values. Furthermore, compared with our method, exhaustive search is more susceptible to this parameter value. Based on above analysis, in terms of running time, we find our method to be more desirable in practice.

3.1.2 Evaluation of accuracy

Here, we observe whether our method misses any joint correlation in practice. To do this, we compare the results found by our method with those of exhaustive search, which guarantees to find all the joint correlations. Notice that, only one parameter, the threshold ϵ, has the potential to affect the results. To this end, we design our experiment as follows. We set the dataset size to 500 genes, and the threshold ϵ' to 0.6. We vary the threshold ϵ from 0.7 to 0.76 at increments of 0.02. For each parameter setting, we repeat the experiment 10 times by randomly selecting a different subset of genes. We report the average and the standard deviation of the total number of joint correlations found by each method. Figure 2d shows the results. A notable observation is that for all ϵ values, our method yields the same number of joint correlations as the exhaustive one. In other words, our method yields 100% accuracy. Moreover, we also observe that the number of joint correlations decreases as we increase the threshold ϵ.



(a) (b) (c) (d)

Figure 2: Performance of our method on a real dataset. (a) Effect of gene set size on running time of filter method and exhaustive method. ϵ and ϵ' are set to 0.75 and 0.6 respectively.(b) Effect of threshold for the joint correlation on running time of filter method. The number of genes is 500 and ϵ' is 0.6. (c) Effect of threshold for the pairwise correlation on running time of filter method and exhaustive method. The running time of the filter method corresponds to the left y scale while that of the exhaustive search corresponds to the right y scale. The number of genes is 500 and ϵ is 0.75. (d) The number of joint correlations obtained by filter method and exhaustive method.

This is expected because large ϵ value enforces more stringent condition for joint correlation.

In summary, the exhaustive method is not practical for large dataset, whereas our method finds gene combinations in a much faster time with the same quality as the exhaustive method.

3.2 Biological significance of the joint correlations

In this section, we explore the biological significance of the joint correlations. For the entire human transcriptome, we set the correlation thresholds ϵ and ϵ' to 0.8 and 0.6, respectively. Using these thresholds, our filter method reports 6,740 jointly correlated gene combinations in one day. As Pearson's correlation is sensitive to outliers, while Spearman correlation is not, we use Spearman correlation to recalculate the joint correlation for those 6,740 combinations, and keep the ones which have correlations greater than or equal to 0.8. Finally, we obtain 482 gene combinations.

Pathways in jointly correlated genes. In our first experiment, we evaluate whether joint correlations reveal functional pathways which cannot be identified through the standard pairwise correlation analysis. We do this in two steps.

(1) We run the Gene Set Enrichment Analysis (GSEA) [19] to find the statistically significant functions and biological processes among 530 unique genes in 482 jointly correlated combinations our method reports. Table 1 shows the significant pathways for these genes.

(2) We obtain the pairwise associated gene sets as follows. First, we build a gene expression correlation network where each node is a gene and an edge links two genes if their Spearman correlation is greater than or equal to the joint correlation threshold ϵ. We then find the maximal cliques on this network [26, 24]. We take the union of genes in all cliques, resulting in totally 2,919 genes. We conduct GSEA on this set. Table 2 presents the result.

Table 1: Summary of GSEA results of jointly correlated gene combinations with comparison to KEGG gene sets.

Description	p-value	FDR q-value
Regulation of actin cytoskeleton	$6.58e^{-10}$	$1.22e^{-7}$
Vascular smooth muscle contraction	$9.27e^{-9}$	$8.62e^{-7}$
Focal adhesion	$6.55e^{-7}$	$4.06e^{-5}$
Leukocyte transendothelial migration	$1.13e^{-6}$	$5.23e^{-5}$
Prostate cancer	$8.64e^{-6}$	$3.21e^{-4}$
Pathways in cancer	$3.01e^{-5}$	$9.33e^{-4}$
MAPK signaling pathway	$6.37e^{-5}$	$1.79e^{-3}$
Dilated cardiomyopathy	$9.25e^{-5}$	$2.15e^{-3}$
Alzheimer's disease	$1.54e^{-4}$	$2.74e^{-3}$
Melanoma	$1.65e^{-4}$	$2.74e^{-3}$

We highlight the pathways common to the two sets in Tables 1 and 2.

We notice that one important pathway that joint correlation finds while the pairwise associated gene groups fails to report is the prostate cancer pathway. This is significant as the samples in our real dataset are taken from tissues with prostate adenocarcinoma. This suggests that joint correlations have the potential to reveal markers for complex disorders like cancer while pairwise correlations can miss due to noise introduced by many false pairwise positive correlations.

Moreover, we observe that joint correlation also yields significant gene enrichment values in other pathways, such as the actin cytoskeleton, Alzheimer's disease and melanoma. The overlapped gene set in actin cytoskeleton has been implicated in prostate cancer [10, 25]. It has been found that there is an inverse association between Alzheimer's disease and cancer presence [3]. Nead *et al.,* [11] demonstrate an association between the use of androgen deprivation therapy (ADT) in the treatment of prostate cancer

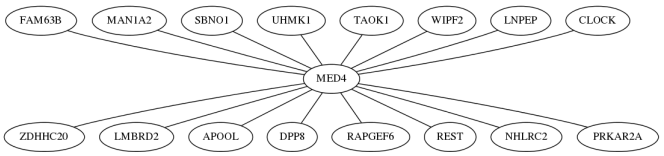
Table 2: Summary of GSEA results of pairwise correlated gene sets with comparison to KEGG gene sets.

Description	p-value	FDR q-value
Ribosome	$2.58e^{-64}$	$4.8e^{-62}$
Focal adhesion	$1.09e^{-40}$	$1.02e^{-38}$
Cytokine cytokine receptor interaeraction	$6.39e^{-31}$	$3.96e^{-29}$
Regulation of actin cytoskeleton	$2.91e^{-24}$	$1.35e^{-22}$
Pathways in cancer	$6.51e^{-24}$	$2.42e^{-22}$
Chemokine signaling pathway	$2.14e^{-23}$	$6.65e^{-22}$
Hematopoietic cell lineage	$4.92e^{-23}$	$1.31e^{-21}$
Ecm receptor interaction	$9.21e^{-22}$	$2.14e^{-20}$
Cell adhesion molecules cams	$9.36e^{-21}$	$1.93e^{-19}$
Oxidative phosphorylation	$2.05e^{-16}$	$3.81e^{-15}$

and an increased future risk of Alzheimer's disease. There also exists an association between prostate cancer and malignant melanoma, that is men with prostate cancer have a significantly increased risk of malignant melanoma [21]. In summary, the results suggest that complex processes and disorders such as prostate cancer and melanoma are governed through nontrivial gene regulations. Joint correlation has the potential to identify such relationships.

Next, we focus on the genes found by our method belonging to the prostate cancer pathway. Some of the genes in this set are SOS1, CREB1 and RB1. Among them, SOS1 is overexpressed in prostate cancer epithelial from African American men [22]. CREB1 is a critical driver of pro-survival, cell cycle and metabolic transcription program, and it has been found that its target gene panels predict prostate cancer recurrence [20]. In castration-resistant prostate cancer, genomic loss of RB1 is the most frequent cell cycle aberration [14]. As a result, RB1 status can be a predictive biomarker to hormonal blockade and cytotoxic taxane therapy.

Finally, we investigate RB1 gene, one of the frequently targeted genes we found. To do this, we build RB1 association network by linking pairs of genes which jointly correlate with it. For instance, assume that g_k denotes a frequently targeted gene for which we build an association network. Also assume that genes g_i and g_j jointly correlate with g_k. We construct two nodes; one for g_i and the other for g_j and connect them with an edge. We repeat this for all such pairs of genes g_i and g_j. Figure 3 presents RB1 association network. We observe that the association network contains one hub node MED4. This hub node jointly regulates RB1 target with many other genes. Thus, these genes can be considered as potential drug targets, as altering them will influence target gene RB1 with a high likelihood. When the transcription of any gene other than the hub node MED4 changes, we expect to have little or



(a) $RB1$

Figure 3: RB1 gene's association network.

no influence on the joint regulation of RB1 as MED4 jointly correlates with many other genes. Alteration of MED4 on the other hand disrupts all potential joint regulations of RB1 and thus has a higher likelihood to alter the transcription of RB1. This is evidenced as it has been found that $RB1^{-/-}$ cells cannot survive in the absence of MED4 [5].

In summary, our proposed method has a great potential to yield biologically significant, yet subtle associations, which cannot be revealed through traditional pairwise association studies, and finding joint correlation of genes is a promising strategy to decipher the functions of genes which are governed through nontrivial interactions.

4 Conclusions

In this paper, we introduced the problem of searching jointly correlated gene combinations. To the best of our knowledge, this is the first computational study in this direction. Finding joint correlation is computationally much harder than the classical pairwise gene correlation problem. The number of combinations for the classical pairwise correlation problem is $O(m^2)$ while that for joint correlation is $O(m^3)$. Exhaustively searching for all jointly correlated gene combinations is infeasible. For example, on a dataset with 800 genes, exhaustive search took more than 10 hours, not to mention that the entire human transcriptome has around 20,000 genes. We developed a novel method for searching such combinations efficiently. This method uses three filters to remove unnecessary gene combinations. Our experiments demonstrated that our method could produce accurate results in a short amount of time. For the entire human transcriptome, our method finished in a day. This shows the efficiency of our methods and its applicability in a real large dataset. We also observed that joint correlations yield biologically significant yet computationally subtle relationships.

Acknowledgment

Publication of this article was funded by UF Health Cancer Center Pilot Project #0012157 and NSF under grant DBI-1262451.

References

[1] The cancer genome atlas homepage. `http://cancergenome.nih.gov`. Accessed: 2016-05-23.

[2] Uri Alon, Naama Barkai, Daniel A Notterman, Kurt Gish, Suzanne Ybarra, Daniel Mack, and Arnold J Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences*, 96(12):6745–6750, 1999.

[3] Maria I. Behrens, Catherine M. Roe, and John C. Morris. Inverse association between cancer and dementia of the alzheimers type. In *Neurodegenerative diseases: From Molecular Concepts to Therapeutic Targets*, pages 111–120. Nova Science Publishers, Inc, New York, 2005.

[4] S Debernardi, S Skoulakis, G Molloy, T Chaplin, A Dixon-McIver, and BD Young. MicroRNA miR-181a correlates with morphological sub-class of acute myeloid leukaemia and the expression of its target genes in global genome-wide analysis. *Leukemia*, 21(5):912–916, 2007.

[5] Catherine Dehainault, Alexandra Garancher, Laurent Castéra, Nathalie Cassoux, Isabelle Aerts, François Doz, Laurence Desjardins, Livia Lumbroso, Rocío Montes de Oca, Geneviève Almouzni, et al. The survival gene med4 explains low penetrance retinoblastoma in patients with large rb1 deletion. *Human molecular genetics*, page ddu245, 2014.

[6] Marcel Dettling, Edward Gabrielson, and Giovanni Parmigiani. Searching for differentially expressed gene combinations. *Genome Biology*, 6(10):R88, 2005.

[7] Elisa Ferrari, Chiara Lucca, and Marco Foiani. A lethal combination for cancer cells: synthetic lethality screenings for drug discovery. *European Journal of Cancer*, 46(16):2889–2895, 2010.

[8] Audrey P Gasch, Paul T Spellman, Camilla M Kao, Orna Carmel-Harel, Michael B Eisen, Gisela Storz, David Botstein, and Patrick O Brown. Genomic expression programs in the response of yeast cells to environmental changes. *Molecular Biology of the Cell*, 11(12):4241–4257, 2000.

[9] Maya Kasowski, Fabian Grubert, Christopher Heffelfinger, Manoj Hariharan, Akwasi Asabere, Sebastian M Waszak, Lukas Habegger, Joel Rozowsky, Minyi Shi, Alexander E Urban, et al. Variation in transcription factor binding among humans. *Science*, 328(5975):232–235, 2010.

[10] Sarah K Martin, Marisa Kamelgarn, and Natasha Kyprianou. Cytoskeleton targeting value in prostate cancer treatment. *American Journal of Clinical and Experimental Urology*, 2(1):15–26, 2014.

[11] Kevin T Nead, Greg Gaskin, Cariad Chester, Samuel Swisher-McClure, Joel T Dudley, Nicholas J Leeper, and Nigam H Shah. Androgen deprivation therapy and future alzheimers disease risk. *Journal of Clinical Oncology*, 34(6):566–571, 2016.

[12] Cancer Genome Atlas Research Network et al. The molecular taxonomy of primary prostate cancer. *Cell*, 163(4):1011–1025, 2015.

[13] Sebastian Nijman. Synthetic lethality: general principles, utility and detection using genetic screens in human cells. *FEBS Letters*, 585(1):1–6, 2011.

[14] Daniel Nava Rodrigues, Gunther Boysen, Semini Sumanasuriya, George Seed, Angelo M De Marzo, and Johann Bono. The molecular underpinnings of prostate cancer: impacts on management and pathology practice. *The Journal of Pathology*, 2016.

[15] Eric E Schadt, John Lamb, Xia Yang, Jun Zhu, Steve Edwards, Debraj GuhaThakurta, Solveig K Sieberts, Stephanie Monks, Marc Reitman, Chunsheng Zhang, et al. An integrative genomics approach to infer causal associations between gene expression and disease. *Nature genetics*, 37(7):710–717, 2005.

[16] RH Segman, N Shefi, T Goltser-Dubner, N Friedman, N Kaminski, and AY Shalev. Peripheral blood mononuclear cell gene expression profiles identify emergent post-traumatic stress disorder among trauma survivors. *Molecular Psychiatry*, 10(5):500–513, 2005.

[17] Charles Spearman. "general intelligence," objectively determined and measured. *The American Journal of Psychology*, 15(2):201–292, 1904.

[18] Joshua M Stuart, Eran Segal, Daphne Koller, and Stuart K Kim. A gene-coexpression network for global discovery of conserved genetic modules. *Science*, 302(5643):249–255, 2003.

[19] Aravind Subramanian, Pablo Tamayo, Vamsi K Mootha, Sayan Mukherjee, Benjamin L Ebert, Michael A Gillette, Amanda Paulovich, Scott L Pomeroy, Todd R Golub, Eric S Lander, et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, 102(43):15545–15550, 2005.

[20] Benjamin Sunkel, Dayong Wu, Zhong Chen, Chiou-Miin Wang, Xiangtao Liu, Zhenqing Ye, Aaron M Horning, Joseph Liu, Devalingam Mahalingam, Horacio Lopez-Nicora, et al. Integrative analysis identifies targetable creb1/foxa1 transcriptional co-regulation as a predictor of prostate cancer recurrence. *Nucleic acids research*, 44(9):4105–4122, 2016.

[21] Frederik B Thomsen, Yasin Folkvaljon, Hans Garmo, David Robinson, Stacy Loeb, Christian Ingvar, Mats Lambe, and Pär Stattin. Risk of malignant melanoma in men with prostate cancer: Nationwide, population-based cohort study. *International Journal of Cancer*, 2016.

[22] Olga A Timofeeva, Xueping Zhang, Habtom W Ressom, Rency S Varghese, Bhaskar VS Kallakury, Kan Wang, Youngmi Ji, Amrita Cheema, Mira Jung, Milton L Brown, et al. Enhanced expression of sos1 is detected in prostate cancer epithelial cells from african-american men. *International journal of oncology*, 35(4):751, 2009.

[23] Chandra L Tucker and Stanley Fields. Lethal combinations. *Nature Genetics*, 35(3):204–205, 2003.

[24] Yun Joo Yoo, Sun Ah Kim, and Shelley B Bull. Clique-based clustering of correlated snps in a gene can improve performance of gene-based multi-bin linear combination test. *BioMed research international*, 2015, 2015.

[25] JS Zhang, A Gong, and CYF Young. ZNF185, an actin–cytoskeleton-associated growth inhibitory LIM protein in prostate cancer. *Oncogene*, 26(1):111–122, 2007.

[26] Chun-Hou Zheng, Lin Yuan, Wen Sha, and Zhan-Li Sun. Gene differential coexpression analysis based on biweight correlation and maximum clique. *BMC bioinformatics*, 15(Suppl 15):S3, 2014.

Analysis of Human Genes with Multiple Functions

Hisham Al-Mubaid
Dept. of Computer Science
Univ. of Houston - Clear Lake. Houston, USA.
hisham@uhcl.edu

Abstract

Genes with multiple functions are very important in an organism as they deliver essential roles. Studying and understanding genes with multiple functions is an important task that can help other problems like gene-disease associations. In this paper, we study gene multifunctionality of all genes in the human genome using the gene ontology and gene functional annotations from GOA database. We propose two gene multifunctionality scoring techniques based on gene annotations from the molecular function *mf* and biological process *bp* aspects. The proposed techniques were examined in estimating and scoring multifunctionality of all human genes, and evaluated the results using four criteria: gene-disease associations; protein-protein interactions *PPI*; gene studies with PubMed publications; and using published known multifunctional gene sets. The evaluation results confirm the validity and reliability of the proposed methods. For example, the proposed methods confirm that multifunctional genes tend to be associated with diseases more than other genes, with significance $p<0.01$, as also proved by previous studies. Moreover, consistent with all previous studies, proteins encoded by multifunctional genes, based on our method, are involved in *PPI* interactions significantly more ($p<0.01$) than other proteins.

1. Introduction

Studying and understanding the function, or set of functions, that a gene is involved in is a central step in functional genomics [1, 2, 16, 17, 26]. In particular, multifunctional genes are important to study as they convey essential roles in an organism [1, 26]. A gene is multifunctional if it is involved in more than one distinct function in human body. Studying and uncovering multifunctional genes is important for various tasks like gene-disease associations, drug discovery, and functional genomics studies.

In this paper, we study human genes in the entire human genome to examine gene multifunctionality and identify the most likely multifunctional genes. Determining if a gene is multifunctional is not a trivial task as many genes can conduct more than one functionality. A gene involved in two functions may not be a multifunctional if the two functions are not distinct (*i.e., not diverse*) enough [1]. In this work, we use a computational methodology to determine whether or not a gene is multifunctional with distinct functions. Specifically, we present two scoring methods based on the functional annotations of the gene from the *Gene Ontology* (GO) for examining gene multifunctionality. We use the GO annotations from the molecular function *mf* and biological process *bp* aspects of *GO*. The proposed gene multifunctionality scoring methods extract and examine all paths between all *mf* and *pb* functions and processes that a gene is annotated with. We examined the proposed methods in scoring and estimating the multifunctionality of all genes in the human genome. We evaluated the results with four different criteria as compared with previous related work in this problem. The four evaluation criteria are: –gene-disease association; –protein-protein interactions *PPI*; –gene studies and *PubMed* publications; and –using published sets of confirmed multifunctional genes. The evaluation results of our proposed methods are encouraging and prove that both scoring methods are valid and reliable indicators of gene multifunctionality. For example, the proposed methods confirm that multifunctional genes tend to be associated with diseases more than other genes, with significance $p<0.01$, as also proved by previous studies. Moreover, consistent with all previous studies, proteins encoded by multifunctional genes, based on our method, are involved in *PPI* interactions significantly more ($p<0.01$) than other proteins.

2. Related Work

One of the most important aspects of multifunctional genes that motivate more work is the gene-disease association. Gene-disease association is significantly higher in multifunctional genes compared to all genes as confirmed by all previous studies in this domain [1–4, 8–10, 26]. Therefor the relationships of diseases and multifunctional genes are signification and proved [1, 10, 26].

A multifunctional gene is a gene that is involved in several functions and activities, including molecular and cellular tasks, inside the cell [1-3, 8]. Pritykin, Ghersi, and Singh (2015) presented a comprehensive study of genome-wide multifunctional genes in human [1]. They found that multifunctional genes are significantly more likely to be involved in human disorders [1]. Also, they found that 32% of all multifunctional genes produced by their method are involved in at least one OMIM disorder, whereas the

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

fraction of other annotated genes involved in at least one OMIM disorder is 21% [1, 7].

Ballouz, Pavlidis, and Gil (2017) studied various gene sets for functional genomics and enrichment [17]. They found that heavily functional genes are highly likely to appear in many genomic study results [21]. They leave it as an 'open question' to biologist to assess if their finding of gene multifunctionality is a true biological property. Khan and Kihara (2016) extracts a domain of features including GO, protein-protein interaction, and more, to classify protein into moonlighting (i.e. multifunctional) versus non-moonlighting proteins [15]. Kim et al. (2017) in their system, *DigSee*, found that genes that interact with more genes in a PPI network are involved in more disease categories than those with fewer neighbors in the protein interaction network [25]

3. Methods for Gene Multifunctionality

The GO is highly regarded as the main source for gene functional information and functional genomics [16, 26]. The proposed gene multifunctionality method is based on the set of annotation terms from the GO for each target gene. The structure of the GO can be used reliably as a function for the relationships among the various functions encoded in the ontology. For example, the path length between two GO terms has been used extensively as a metric in computing semantic similarity between genes [16, 20, 26]. A semantic similarity *measure* is a function that estimates the similarity between two genes or two GO terms as a numeric value [19, 20]. Moreover, many gene similarity measures use the depth of the lowest common subsumer (LCS) in computing gene similarity [19, 20]. In our previous work, we investigated and explained the relationship between GO annotation terms of a gene and gene-disease relationship [16].

In this paper, we propose and present two methods derived from the gene ontology for scoring gene multifunctionality. Typically, the similarity between two genes is computed as a function of the similarity between their annotation terms from the Gene Ontology (GO) using the *mf* (or the *bp*) aspect. That is, the similarity $Sim_g(g_1, g_2)$ between two genes g_1 and g_2 can be a similarity function between the annotations of g_1 and g_2:

$$Sim_g(g_1, g_2) = Sim_t(t_{1i}, t_{2i}) \ldots\ldots\ldots\ldots\ldots (1)$$

where $Sim_g(g_1, g_2)$ is the similarity between genes $g_1$and g_2; and $Sim_t(t_{1i}, t_{2i})$ is the similarity between GO terms t_{1i}, t_{2i} annotating $g_1$and g_2 respectively.

The gene ontology consists of 3 aspects: Molecular Function *mf*, Biological Process *bp* and Cellular Component *cc*. Each one of these aspects {*mf*, *bp*, *cc*} is a complete ontology in itself [6, 16, 20, 26]. For gene multifunctionality it is normal to rely only on *mf* and *pb* aspects.

Let $MaxPL_f(g_x)$ be the *maximum* path length between all pairwise *mf* annotation terms of gene g_x; that is:

$$MaxPL_f(g_x) = \max_{t_x, t_y \in GOT_f(g_x)} PL(t_x, t_y) \quad \ldots\ldots\ldots (2)$$

where $PL(t_x, t_y)$ is the *shortest* path length between the two GO terms t_x and t_y, and $GOT_f(g_x)$ is the set of all GO *mf* annotation terms (*annotations*) of gene g_x. For example, in Figure 1, there are two different paths shown between GO:0000001 and GO:0006996 one of them is of length 2 (through GO:0048308) and the second path is of length 3 (through the two GO terms GO:0048311 and GO:0007005). The multifunctionality of a gene increases with the increase in the distinctiveness (*i.e.*, diversity) of the functions that the gene in involved in [1, 3]. The path length between two *mf* (or *bp*) annotation terms of a target gene can be utilized as an indicator of the *distinctiveness* of the functions that the gene is part of. Based on this, we employ $MaxPL_f$ as defined in equation (2) as a multifunctionality scoring method based on the max path length between the *mf* annotation terms. Likewise, we compute multifunctionality score based on *bp* annotation terms as:

$$MaxPL_p(g_x) = \max_{t_x, t_y \in GOT_p(g_x)} PL(t_x, t_y) \quad \ldots\ldots\ldots (3)$$

where $GOT_p(g_x)$ is the set of all *pb* annotations of gene g_x. In the biological process aspect (*bp*) of GO, each annotation term is basically a node in the ontology graph and is a biological functionality upheld by certain genes. When two *bp* annotation terms (i.e., *graph nodes*) are far apart with relatively large path length between them then we can consider that these two terms represent two distinct (*diverse*) biological functionalities. That is, our hypothesis is that, two highly far apart *bp* annotation terms can be considered as two distinct biological processes. Therefore, a gene annotated with two such terms can be considered as multifunctional.

The computations of multifunctionality scores with *mf* annotations for human genes go through the algorithm shown in Figure A1. This algorithm explains the steps for the *mf*-based multifunctionality scoring, and the *bp*-based scoring is computed similarly by replacing *mf* annotations with *bp* annotations. For each gene, we extracted all its annotation terms from the Gene Ontology Annotation *GOA* database for human [26].

By considering only *mf* annotations we found a total of ~35,800 genes annotated with at least one *mf* terms. Overall, there are ~4.3 *mf* terms annotated per gene. By considering *bp* annotations in *GOA*, there is on average 5.2 *bp* terms per

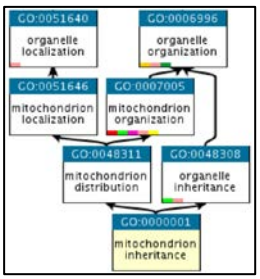


Figure 1: a small part of the *GO*.

Algorithm 1: Compute multifunctionality scores for all genes

Input: -*GOA_human*: set of all human gene annotations.
-*GO.obo*:set of all gene ontology annotation terms with their parents
Output: -Set $\{MaxPL_f(g_x)\}g_x \in G$: multifunctionality score for every human gene based on *mf* annotations.
Algorithm:
(1) Create the set G
1a) G = ∅ : let G be the set of all genes annotated in *GOA_human*
1b) For each annotated gene g_i from the set *GOA_human*:
i) $G = G \cup g_i$: add g_i to G
(2) Create the set MF
2a) MF = ∅ : let MF be the set of all *mf* annotation terms in *GO.obo*
2b) For each *mf* annotation term t_i in *GO.obo*:
i) $MF = MF \cup t_i$:add t_i to MF along with its parents
(3) Create the set *GOA_human_mf*
3a) Extract all *mf* annotations from *GOA_human* and add them to *GOA_human_mf*
(4) For each gene g_x in the set G
4a) Extract the set $GOT_f(g_x)$ of all annotations of g_x from *GOA_human_mf*
4b) Set $MaxPL_f(g_x) = 0$
4b) If $|GOT_f(g_x)| < 2$ go to step (4)
4c) For each pair t_i, t_j of annotation terms in $GOT_f(g_x)$:
i) Compute the shortest path length $PL(t_i, t_j)$ between pair t_i, t_j
ii) If $PL(t_i, t_j) > MaxPL_f(g_x)$ then set $MaxPL_f(g_x) = PL(t_i, t_j)$

Figure A1: Algorithm for $MaxPL_f()$ of all human genes

gene with a total of ~35,700 genes annotated with at least one *bp* term. Among all genes with *mf* annotations (~35,800 genes) in GOA database, almost 42% of them (or 15,142 genes) are annotated with only one *mf* annotation term. Clearly, each gene with only one *mf* term will have $MaxPL_f = 0$ (and similarly for *bp*). Therefore, in *mf* we have 42% of the genes do not count in the computations of the multifunctionality scoring method $MaxPL_f$. In the human annotations in *GOA*, ~80% of the genes (= *28,904 genes*) have 4 or fewer *mf* terms. Thus, we extracted all genes from the *GOA* human annotation database. We computed the maximum path length among all terms for every gene as per our proposed technique. For evaluation, we would like to verify the reliability of our multifunctionality scoring techniques, $MaxPL_f$ and $MaxPL_p$, in estimating the whether or not a gene is multifunctional. We could not find any gold standard dataset to evaluate our methods. So, we used four criteria for multifunctionality [1, 26, 15]. These four criteria are: (1) Gene-disease association is more in multifunctional genes compared with other non-multifunctional genes; (2) Multifunctional genes are more evolutionary conserved; (3) Multifunctional genes tend to be highly studied with relatively higher number of publications; and (4) Using previously tested and published multifunctional gene sets as criteria to test our method.

We analyzed all human genes having *mf* annotations using proposed $MaxPL_f(g_x)$ multifunctionality scoring system. After computing $MaxPL_f(g_x)$ value for each gene, we grouped all genes into clusters of 1000 genes in each cluster after being sorted based on $MaxPL_f(g_x)$; *see Table 1*.

Table 1: Average $MaxPL_f$ with clusters of 1000 genes in each cluster.

After sorting all genes based on $MaxPL_f()$ (*descending order*)	**mean $MaxPL_f()$**
Top 1000	13.987
1001 – 2000	12.189
2001 – 3000	11.250
3001 – 4000	10.427
4001 – 5000	9.576
5001 – 6000	8.487
6001 – 7000	7.505
7001 – 8000	6.336
8001 – 9000	2.030
9001 – 10000	3.189
10001 – 11000	0.880
11001 – 12000	0.498
Lowest 1191	0

For example, as shown in Table 1, the top 1000 genes have an average $MaxPL_f(g_x)$ of 13.99 whereas the next cluster (next 1000 genes) have $MaxPL_f$ average of 12.19.

Criteria 1. *Gene-disease associations:*

One of the most important criteria of multifunctionality of a gene is its disease associations [1-3, 16, 25, 26, 27]. That is, multifunctional genes are more highly likely to be associated with human diseases than non-multifunctional genes [1, 16, 25]. We analyzed all human genes from the *GOA* database and from *OMIM morbid map* for disease information. We wanted to investigate if the number of phenotypes, according to *morbid map*, exhibits any meaningful relationship with our multifunctionality scoring $MaxPL_f()$. The results are in Table 2 and illustrated in Figure 2. As it is shown in both Table 2 and Figure 2, as the $MaxPL_f$ increases the average number of associated phenotypes increases; thus there is a clear strong correlation between $MaxPL_f$ and average number of phenotypes. Hence, our $MaxPL_f$ is a reliable indicator of multifunctionality of genes. Next, we examined the behavior of $MaxPL_f$ with the increase of phenotypes for all human genes and the results are illustrated in Figure 3.

We repeated the same evaluation for *MaxPLp*, that is, using *bp* annotations (instead of *mf* annotations). Table 3 and Figure 4 show the correlation between *MaxPLp* and average number of phenotypes for all human genes.

Next, we examined *MaxPLp* for each group of genes associated with the same number of phenotypes and the results are in Table 4. For example, there are 2572 genes associated with only one phenotype and their average *MaxPLp* is 11.22 whereas the group of genes associated with exactly two phenotypes (648 genes) have an average *MaxPLp* of 12.33; Table 4. We mention here that groups of genes associated with 7 or more phenotypes are very small and do not affect the results. For example, there are only 11 genes associated with 7 disease, and only 7 genes associated with 8 diseases. Figure 5 shows the relationship of the average number of phenotype for each value of $MaxPL_p$.

Table 2: Average number of phenotypes for genes with each value of $MaxPL_f$

$MaxPL_f$	No. of genes	Avg. of No. of phenotypes
0	2630	0.23
1	244	0.26
2	490	0.27
3	467	0.37
4	557	0.36
5	578	0.36
6	890	0.30
7	831	0.41
8	1018	0.44
9	911	0.49
10	1148	0.46
11	1182	0.52
12	1057	0.49
13	512	0.68
14	475	0.68
15	126	0.77
16	53	0.66
17	10	1.10
18	11	0.73
19	1	0.00

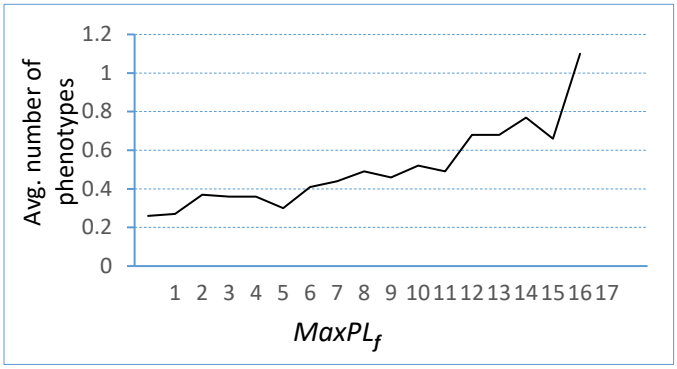


Figure 2: from Table 2 above: average number of phenotypes increases as a function of $MaxPL_f$

Criteria 2. *Protein-protein interactions:*
Multifunctional genes are typically involved more than normal in protein-protein interactions (PPI) [1, 22, 23, 25, 17]. We used this criterion in evaluating our proposed multifunctionality methods. We retrieved and compiled *PPI* data from the Hippie database [23-24]. The obtained data include *PPI* data involving ~14,800 genes with a total of ~250K experimentally documented P-P interactions [22–24]. We analyzed the average number of PPI's that a gene involved in with respect to both $MaxPL_f$ and $MaxPL_p$ and there is a clear relationship as illustrated in Figure 6. These results prove again that the proposed methods are in line and consistent with this criteria for gene multifunctionality.

Criteria 3. *Using PubMed publications as indicator of highly studied:*
It has been shown that multifunctional genes are highly studied genes and have relatively more publications in the biomedical literature [1, 21]. So, we use publication counts of genes as a criteria of multifunctionality. That is, multifunctional genes tend to have relatively higher number of publications compared to all genes. We relied on *PubMed* since it is the most comprehensive repository of biomedical literature with more than 24 million citations and references to articles (*with abstracts, and some with full texts*). We analyzed number of publications related to each gene in *PubMed* as it is published by NCBI/PubMed and freely available with file name: *gene2pubmed.gz*, (*link: ftp://ftp.ncbi.nlm.nih.gov/gene/DATA;downloaded Sept.2017*). We examined genes with our multifunctionality scores versus number of publications. The analysis results shows a clear straightforward proportionality between number of publications and both scoring methods ($MaxPL_f$ and *MaxPLp*) for all human genes as illustrated in Figure 7.

Criteria 4. *Using published multifunctional genes:*
We retrieved two lists of experimentally tested and known multifunctional genes [1].
Source 1: http://moonlightingproteins.org/proteins/, which includes 361 moonlighting proteins (*74 for human*).
Source 2: http://wallace.uab.es/multitask/ which includes 288 proteins (*88 of them for human*).
This test was not reliable as we are considering only 162 (*74 from set1 and 88 from set2*) human genes (*out of ~35000 annotated genes*); however, these genes exhibit higher $MaxPL_f$ and $MaxPL_p$ values than expected by chance with significance ($p<0.01$) confirming multifunctionality.

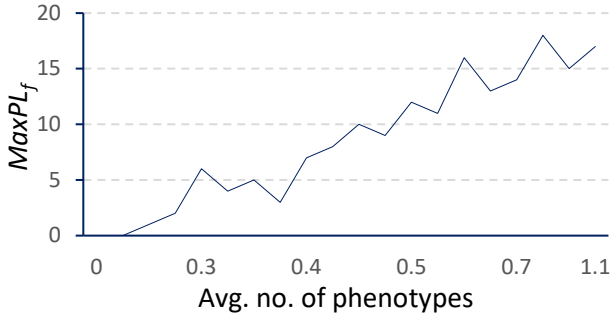


Figure 3: This figure shows that the $MaxPL_f$ of genes is directly proportional with average no. of phenotypes.

Table 3: For each value of *MaxPLp* this table shows how many genes and the average number of phenotypes

MaxPLp	No. of genes	Avg. of No. of phenotypes
0	1399	0.15
1	71	0.32
2	141	0.21
3	158	0.19
4	204	0.26
5	266	0.21
6	356	0.35
7	497	0.31
8	578	0.31
9	733	0.31
10	919	0.35
11	1319	0.32
12	1490	0.36
13	1541	0.45
14	1512	0.48
15	1166	0.58
16	725	0.64
17	455	0.85
18	217	0.76
19	113	0.73
20	13	1.00
21	2	3.00

Table 4: The average value of *MaxPLp* for seven groups of genes where each group have the same number of phenotypes.

No. of phenotypes	No. of genes	Avg. of *MaxPLp*
0	10234	10.09
1	2572	11.22
2	648	12.33
3	226	12.96
4	84	13.67
5	51	13.86
6	29	15.45

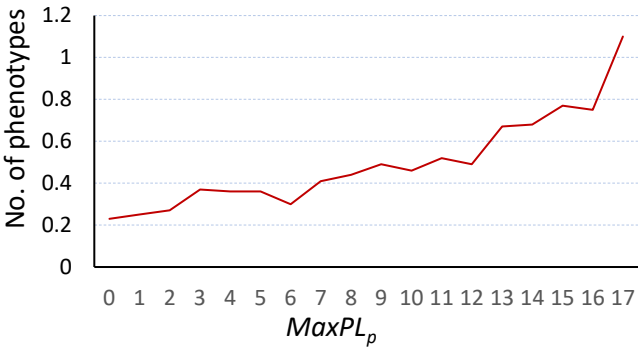


Figure 4: The relationship between $MaxPL_p$ and number of diseases for all human genes.

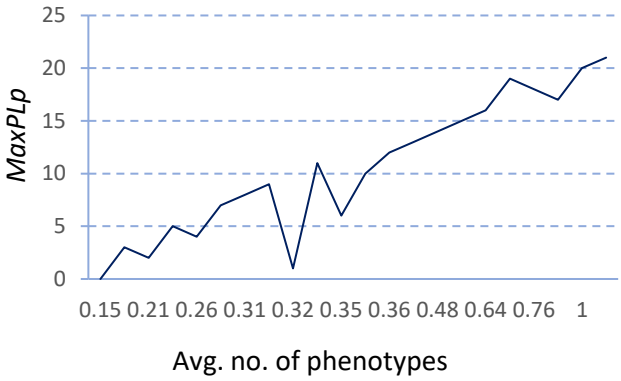


Figure 5: The relationship of the average number of phenotype for each value of *MaxPLp*

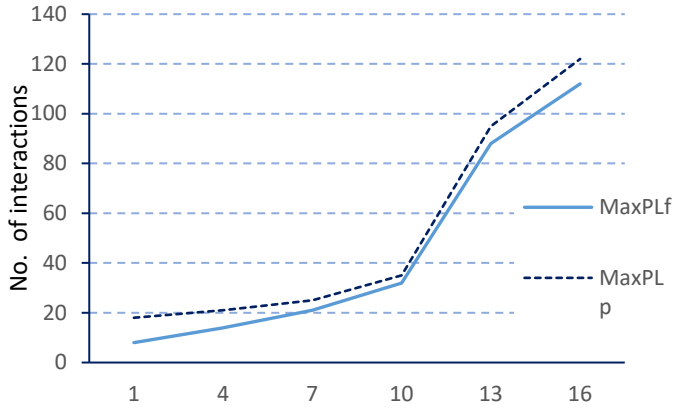


Figure 6: Number of protein-protein interactions increases as the gene multifunctionality score increase.

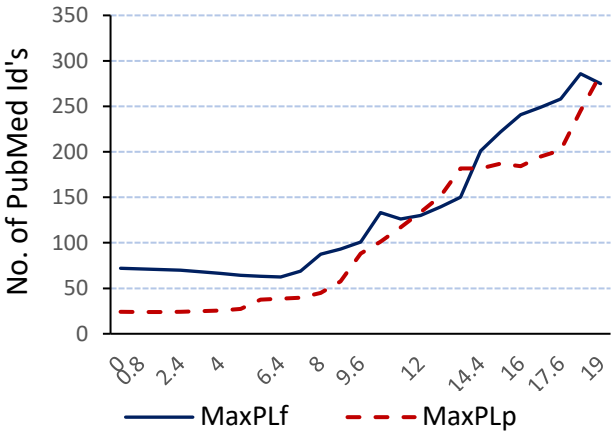


Figure 7: Analyzing number of *PubMed* publications against multifunctionality scores with both *MaxPLf* and *MaxPLp* for all human genes show a direct proportional relationship.

4. Discussion

The results in Table 2 (and Figure 2) confirm that there is a direct proportional relationship between gene multifunctionality scores and the number of diseases associated with gene. For example, genes with $MaxPL_f$ =2 (490 genes) have on average 0.27 associated diseases whereas genes with $MaxPL_f$ =3 (467 genes) are on average associated with 0.37 diseases; and this is significant $p<0.01$ (using hypergeometric test). Also, the average number of phenotypes for 1182 genes with $MaxPL_f$ of 11 is 0.52, and when the $MaxPL_f$ increases to 13 the average number of phenotypes increases to 0.68 (Table 2) which is significant result ($p<0.01$).

We observed that for genes with one phenotype (2442 genes) the average $MaxPL_f$ is 7.40 whereas for those genes with 2 phenotypes the average MaxPLf increases to 8.39 and this is significant with $p<0.01$; as follows:

No. of phenotypes	No. of genes	Avg. $MaxPL_f$
1	2442	7.40
2	619	8.39

Similarly, for *MaxPLp*, we see that the 10234 genes associated with 0 phenotypes have *MaxPLp* average of 10.09, and for the genes associated with one phenotype (*2572 genes*) the value of *MaxPLp* increases to 11.22 (as shown below) which is significant $p<0.01$.

No. of phenotypes	No. of genes	Avg. MaxPLp
0	10234	10.09
1	2572	11.22

Regarding number of publications, *criteria 3*, we confirmed that as our multifunctionality score of a gene tend to increase, the number of *PubMed* publications related to the gene also increases as illustrated in Figure 7. We should mention here that higher number of publications implies that the gene is highly studied [1]. One of the main reason of being highly studied is the gene is highly likely associated with one or more diseases. We should mention here that there are genes with fairly high number of publications but with low (≤ 7) multifunctionality score for which reason we relied on the aggregate averages. For example, considering genes with *MaxPLp* of 12; their average number of *PubMed* publications is ~133; when we increase the score to 14 the average increases to ~181 and this is significant ($p<0.01$). Finally, if we consider a multifunctional every gene with $MaxPLp \geq 15$, we get 2691 multifunctional genes (genes having *MaxPLp* of 15 or more). Among these 2691 genes, we found 46% (or 1231 genes) of them are also *mf* multifunctional with *mf* annotations only using threshold T_f=10 ($MaxPL_f \geq 10$), and this is significant $p<0.01$ with hypergeometric test.

5. Conclusion

For future studies of this research, we would like to investigate the number of maximum path lengths between the annotations of the target gene. For instance, consider the following case: If $MaxPL_f(g_x) = 16$ and $MaxPL_f(g_y) = 16$ but $NoMaxPL_f(g_x) = 1$ while $NoMaxPL_f(g_y) = 3$, where $NoMaxPL_f(g_x)$ is the number of paths with max length. In this case, both genes g_x and g_y have $MaxPL_f$ of 16 but this $MaxPL_f$ of 16 occurs and repeated three times in gene g_y and once in g_x making g_y more multifunctional. In addition, in the future work in this direction, we would like to investigate a multifunctionality score (*mfs*) that relies on both *mf* and *bp* annotations normalized by some maximum value, *e.g. fp*, as follows:

$$mfs(g_x) = \frac{MaxPLf(g_x) + MaxPLp(g_x)}{fp}$$

References

[1] Y. Pritykin, D. Ghersi, M. Singh. Genome-Wide Detection and Analysis of Multifunctional Genes. PLOS Computational Biology, October 5, 2015

[2] Van de Peppel J, Holstege FCP (2005) Multifunctional genes. Molecular Systems Biology 1: 1–2. doi: 10.1038/msb4100006, 2005.

[3] I. Khan, Y. Chen, T. Dong, Hong X, Takeuchi R, et al. (2014) Genome-scale identification and characterization of moonlighting proteins. Biology Direct 9: 30. doi: 10.1186/s13062-014-0030-9 PMID: 25497125, 2014.

[4] Becker E, Robisson B, Chapple C, Guénoche A, Brun C (2012) Multifunctional proteins revealed by overlapping clustering in protein interaction network. Bioinformatics 28: 84–90. PMID: 22080466, 2012.

[5] Ashburner et al. Gene Ontology: tool for the unification of biology (2000) *Nat Genet* 25(1):25-9. Online at Nature Genetics.

[6] The Gene Ontology Consortium. Gene Ontology Consortium: going forward. (2015) *Nucl Acids Res* 43 Database issue D1049–D1056. Online at Nucleic Acids Research, 2015.

[7] Online Mendelian Inheritance in Man, OMIM. McKusick-Nathans Institute of Genetic Medicine, Johns Hopkins University (Baltimore, MD), May 2016. World Wide Web URL: http://omim.org/

[8] Hernández S, Ferragut G, Amela I, Perez-Pons J, Pinol J, et al. (2014) MultitaskProtDB: a database of multitasking proteins. Nucleic Acids Research 42: D517–D520. doi: 10.1093/nar/gkt1153. pmid:24253302, 2014.

[9] Mani M, Chen C, Amblee V, Liu H, Mathur T, et al. (2015) Moonprot: a database for proteins that are known to moonlight. Nucleic Acids Research 43: 2015.

[10] A. Day, J. Dong, V. A. Funari, B. Harry, S.P. Strom, D.H. Cohn, and S. F. Nelson. Disease Gene Characterization through Large-Scale Co-Expression Analysis. PLoS ONE Vol.4, Issue 12, 2009.

[11] J. Gillis, P. Pavlidis (2013) Assessing identity, redundancy and confounds in gene ontology annotations over time. Bioinformatics 29: 476–482. doi: 10.1093/bioinformatics/bts727, 2013.

[12] M. Salathe M, Ackermann M, Bonhoeffer S The effect of multifunctionality on the rate of evolution in yeast. Molecular Biology and Evolution 23, 2006.
[13] W.T. Clark, Radivojac P (2011) Analysis of protein function and its prediction from amino acid sequence. Proteins: Structure, Function, and Bioinformatics 79, 2011.
[14] J. de Peppel and F. CP Holstege. Multifunctional genes. Molecular Systems Biology 1: 1-2, 2005.
[15] I. K. Khan, and D. Kihara. Genome-scale prediction of moonlighting proteins using diverse protein association information. Oxford 2016.
[16] H. Al-Mubaid et. al. Assessing Gene-Disease Relationship with Multifunctional Genes Using GO. Proc. of IEEE AICCSA 2016.
[17] Gillis J., Pavlidis P. The impact of multifunctional genes on 'guilt by association' analysis. PloS One. Vol.6 no.2, 2011; 6:e17258.
[18] NCBI. Clearing Up Confusion with Human Gene Symbols & Names Using NCBI Gene Data, USA.
[19] K. Glass and M. Girvan. Finding New Order in Biological Functions from the Network Structure of Gene Annotations. PLoS Comput Biol. 11(11): 2015.
[20] A. Nagar and H. Al-Mubaid. A Hybrid Semantic Similarity Measure for Gene Ontology Based On Offspring and Path Length. Proc. of IEEE CIBCB-2015 IEEE Conf. on Computational Intelligence in Bioinformatics and Comp. Biology, 2015. DOI: 10.1109/CIBCB.2015.7300290.
[21] Sara Ballouz, Paul Pavlidis, and Jesse Gillis. Using predictive specificity to determine when gene set analysis is biologically meaningful. Nucleic Acids Res. 2017; 45(4): e20. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5389513/
[22] H. Zhou and J. Skolnick. A knowledge-based approach for predicting gene–disease associations. Bioinformatics. 2016; 32 (18): 2831–2838.
[23] Schaefer MH, Fontaine JF, Vinayagam A, Porras P, Wanker EE, Andrade-Navarro MA. HIPPIE: Integrating protein interaction networks with experiment based quality scores. PLoS One. 2012; 7(2):e31826.
[24] Hippie PPI database http://cbdm-01.zdv.uni-mainz.de/~mschaefer/hippie/
[25] J. Kim, J-j Kim, and H. Lee. An analysis of disease-gene relationship from Medline abstracts by DigSee. Scientific Reports; 2017. 7: 40154. DOI: 10.1038/srep40154
[26] H. Al-Mubaid et. al. Determining Multifunctional Genes and Diseases in Human Using Gene Ontology. Proceedings of 9th Int'l Conf on Bioinformatics and Computational Biology BICOB-2017, Honolulu, USA, March 2017.
[27] Singh-Blom UM, Natarajan N, Tewari A, Woods JO, Dhillon IS, et al. (2013) Prediction and Validation of Gene-Disease Associations Using Methods Inspired by Social Network Analyses. PLOS ONE 8(9), 2013.

Selection of Informative Genomic Regions for Closely Related Isolates and Construction of their Phylogeny

Anindya Das and Xiaoqiu Huang
Department of Computer Science, Iowa State University
Ames, IA, 50010, USA
(anindya, xqhuang)@iastate.edu

Abstract

Building a phylogenetic tree for a number of species is a very important step towards understanding the evolutionary history of those species. If some of these species are hundreds of times more closely related than others, then the subtree(s) (or clade) of more closely related species may have a very low level of resolution and may not accurately represent their evolutionary history. Identification of the informative regions containing important variations in the genome of those species is important in constructing their evolutionary history. Here we introduce a novel approach for selecting informative regions in an effort to construct a phylogenetic tree of these closely related subspecies (isolates) with high resolution. This approach is based on the observation that the likelihood of informative columns are sensitive to changes in the tree topology. We also propose a method for identifying clades with low resolution in the tree using branch lengths and likelihoods. We show that reconstructed phylogenies from the informative columns (identified by our method) are more accurate for the closely relates isolates than the phylogenies constructed from the whole alignment.

keywords: Maximum Likelihood, Phylogenetic Tree, Resolution of Tree, Informative Regions of Alignment.

1 Introduction

Inference of phylogeny from nucleotide or protein sequences of a number of species has been studied extensively for a long time. Distance methods (e.g. Neighbor-Joining [17]) and maximum parsimony methods [20] have been applied for building phylogenetic trees. As distance methods are more efficient, trees constructed by these methods are often used as an initial tree in other methods. Maximum likelihood [3] and Bayesian inference [16] are two commonly used methods where a mathematical model of substitution like Jukes-Cantor (JC69) [9], F84 [4] or the General Time-Reversible model (GTR) [21] is applied. Multiple studies [7, 10, 14] have shown that maximum likelihood methods produce accurate trees. Therefore, maximum likelihood trees are frequently used to derive the phylogenetic relationships among species. Programs like PhyML [6], RAxML [19], and MEGA [11] have been developed to construct maximum likelihood trees from sequence data.

A tree constructed by the maximum likelihood approach is built from a multiple sequence alignment. Sometimes one or more groups of isolates contain all the variations in a small region of the whole alignment. As a result, those isolates are so close that their phylogeny is considered unresolved in one study [18]. It has been shown that small regions or few columns of the alignment can significantly affect the resolution and topology of a particular clade [18, 2]. Therefore, identifying those small regions in the alignment is helpful in constructing their phylogeny. Determining the influence of an outlier site (or column) [2] on the phylogeny has been studied by removing the column from the alignment and by constructing topology and likelihood from the resulting alignment. Tens of thousands of microbial genome sequences are publicly available. Some of them have an extremely low rate of single nucleotide polymorphism (SNP), e.g., one SNP in a million base pairs [1]. On a data set of 101 whole genome sequences with low pairwise SNP rates, existing programs were able to infer up to 71% of the clade structure [1]. To the best of our knowledge, none of the exsiting programs has been designed to handle whole genome sequences with both high and low SNP rates.

In a common approach to constructing a phylogenetic tree of isolates, multiple gene datasets for the isolates are selected based on human knowledge, each gene dataset is used to build a gene tree, and the gene trees are reconciled to obtain a specie tree [5]. This approach has been used for isolates that are not highly similar. Note that isolates with an extremely low SNP rate are identical in sequences over most loci. We take a complementary approach by eliminating human involvement in deciding which genome regions are selected. Our approach uses computational and

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

statistical techniques to decide which genome positions are informative for which part of the species tree. Below we describe a method for identifying a group of isolates with low resolution using informative columns. Then we reconstruct the phylogeny (cladistic relationship) of those closely related isolates using informative columns. Results from our method on simulated data show that the trees constructed from the whole alignment are less accurate than the trees constructed from informative alignment columns for the closely related isolates.

2 Methodology

We address the problem of constructing a phylogenetic tree of isolates by building an unrooted bifurcating tree [3] from a multiple alignment of genome sequences of the isolates. An important feature about the multiple genome alignment is that the rate of substitution between some genome sequences can be hundreds of times smaller than that between other genome sequences in the alignment. Below we first describe a method for selecting informative regions for a group of isolates and then we present a method for detecting subtrees of isolates with low resolution.

2.1 Detection of Informative Regions

There are well-known programs for building a tree from an alignment using the maximum likelihood approach. Our method begins with one of such programs. First, a maximum likelihood tree T is created using RAxML from a given multiple sequence alignment S. The program seeks to find a tree with maximum likelihood. The likelihood of a tree with respect to the given alignment is the product of the likelihood with respect to each column in the alignment. The likelihood value of each column of the alignment can also be computed using this program for any given tree. We use the following idea to evaluate the importance of a column:

If a column is informative for a particular clade, then there must be a significant change in the likelihood value of that column with respect to a *nonrepresentative* topology for that clade.

We use the following two methods for generating these nonrepresentative topologies:

(1) Performing the nearest neighbor interchange (NNI) operation [12, 22] on any edge (in the clade) connecting two non-leaf nodes

(2) Randomly permuting the isolates so that it creates a random distribution of the isolates in the clade

Let N be the number of isolates in a clade. We can generate 2 nonrepresentative topologies by applying the NNI operation on each of the $N-3$ edges connecting two non-leaf nodes. We also generate k ($k \leq N$) random permutations of those isolates. Then we find the maximum deviation of likelihood for a column with respect to these nonrepresentative topologies.

An example of deviation of likelihood values for one informative column and one noninformative column is shown in Table 1. Here, T is the maximum likelihood tree (representative topology) constructed from an alignment S and T_{NNI} is the nonrepresentative topology generated by an NNI operation on one edge of T. We can observe that the deviation of likelihood of the informative column is 19.56 times that of the noninformative column with respect to this NNI operation.

Table 1: Deviation of Likelihood for Alignment S

	Log Likelihood	
	Informative Column	Noninformative Column
T	-5.661510	-1.867485
T_{NNI}	-7.524911	-1.962215
Deviation	1.853401	0.09473

Now we need to check whether this deviation is statistically significant or not. We introduce the concept of noise alignment to make this decision. The noise for a particular column is generated through a random permutation (different from the alignment S) of the rows. Thus we choose a random permutation of rows for each column to form a noise alignment S'. We do not expect a large difference between the deviation of likelihood values for informative and non-informative columns with respect to S'.

For example, we can look at the deviation of likelihood values in Table 2 for the informative and noninformative columns (presented in Table 1) with respect to noise alignment S'. Both T and T_{NNI} are the same trees mentioned in Table 1. Here, the difference between the deviation of likelihood values for the informative and noninformative columns is not very large. Moreover, the deviation of likelihood for the noninformative column with respect to S shown in Table 1 is also close to the values of deviation shown in Table 2. We formulate the following criteria to select informative columns based on these observations:

If a column generates a significantly higher deviation in likelihood value than the average deviation of the columns when they are subjected to a *noise* alignment (i.e. the p-value of the deviation of a column computed from the distribution of deviations with respect to noise is less than the significance level α), then we classify that column to be informative for a clade.

Table 2: Deviation of Likelihood for Noise Alignment

	Log Likelihood	
	Informative Column	Noninformative Column
T	-6.974545	-1.839750
T_{NNI}	-6.953124	-1.795106
Deviation	0.021421	0.044644

The key steps of our algorithm for selecting informative columns for N taxa from an alignment S are summarized below. Here, T is the maximum likelihood tree constructed from S and L is the number of columns in S.

(1) A set T_n of nonrepresentative phylogenies is constructed, which contains k trees generated by random permutation of N taxa and $2 * (N - 3)$ trees generated by NNI operation on T.

(2) A noise alignment S' is generated from S.

(3) The maximum deviation of likelihood value of each column over $|T_n|$ nonrepresentative phylogenies with respect to S' is computed. These values of maximum deviation for each column form the noise distribution.

(4) The maximum deviation of likelihood value of each column over $|T_n|$ nonrepresentative phylogenies with respect to S is computed. If the maximum deviation of likelihood of a column is statistically significant with respect to the noise distribution, then this column is selected as informative.

Here, steps 3 and 4 are the dominating steps of the algorithm. In both steps, we compute the deviation for each of the L columns in the alignment S with respect to $|T_n|$ nonrepresentative topologies generated from T. Computing the likelihood value of a column with respect to a topology t requires a traversal over t, which requires $O(N)$ operations because of $2 * N - 1$ nodes in the tree. Now, $|T_n| = 2 * (N - 3) + k$, where we choose $k \leq N$. Therefore, the complexity of this algorithm is $O(L * N^2)$. As the complexity of this algorithm is linear in terms of the number of columns in the alignment, this method is efficient on large alignments.

2.2 Selection of Clades with Low Resolution

We use a likelihood based method for selecting clades with low resolution in the tree T. For each edge connecting two non-leaf nodes with at least 4 taxa as each of their descendants, we can form two components by removing the edge. We consider each component as a clade. Then for each clade, if the number of informative columns (identified by the method described earlier) is less than a certain threshold, then that clade is considered to have low resolution in the tree.

We have also used a heuristic based on the distribution of branch lengths to make this step faster. If the average branch length of one clade is less than the average branch length of the tree, then we consider this clade to be a candidate for having low resolution. Thus we can avoid finding informative columns for every clade in the tree.

2.3 Reconstruction of the Phylogeny of Closely Related Isolates

After the identification of closely related isolates with low resolution, we form a multiple sequence alignment S_c for those closely related isolates. Then we construct a phylogeny T_c from S_c using RAxML. Then we find informative columns and construct a phylogeny T_{c_i} (tree of closely related isolates from informative columns).

3 Results

Our results are divided into two parts. First, we show that our method produced a tree with high resolution from a large sequence alignment of 11 *Fusarium* isolates. Then we show that our method produced a more accurate tree of closely related isolates on simulated data. We have used 0.01 for the significance level α on both real and simulated data.

3.1 Real Data

We used a sequence alignment of 11 *Fusarium* isolates [8] that cause diseases in soybean. The maximum likelihood tree in Figure 1 was constructed by RAxML.

Here the clade containing 4 *F. virguliforme* isolates has low resolution. Our method for selecting clades with low resolution correctly identified this group of *F. virguliforme* isolates. Then we applied the likelihood based column selection method to identify 319 informative columns from the alignment of 137,718 columns. Then we reconstructed their phylogeny using these columns with high resolution (Figure 2).

3.2 Simulated Data

We used the Seq-Gen program [13] to generate a large number of sequence alignments for 8 species (or taxa) from the tree with two groups in Figure 3. Each

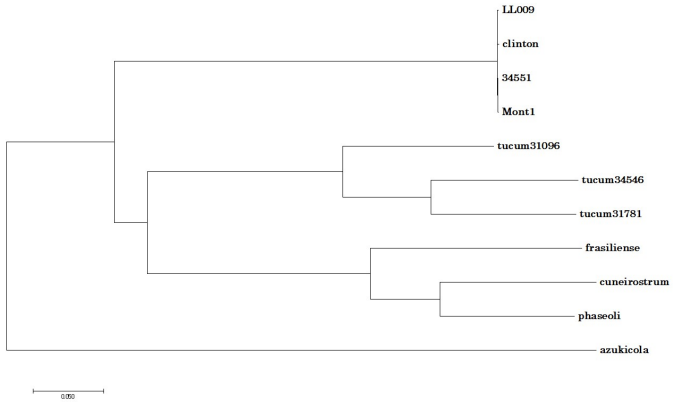


Figure 1: The maximum likelihood tree of 11 *Fusarium* Isolates from an alignment of 137,718 columns

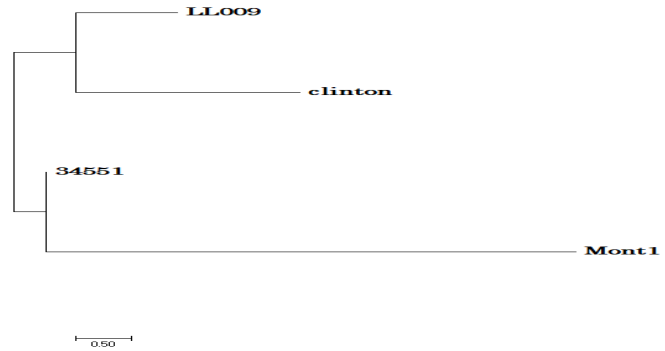


Figure 2: The maximum likelihood tree of 4 *F. virguliforme* Isolates from an alignment of 319 informative columns

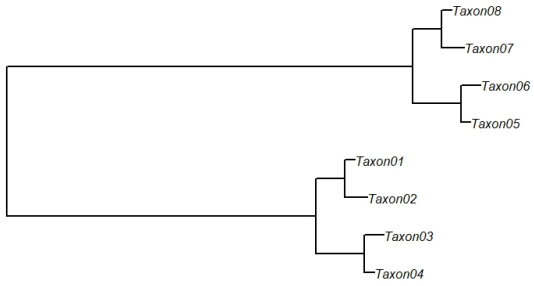


Figure 3: The topology used to generate the sequences for all cases

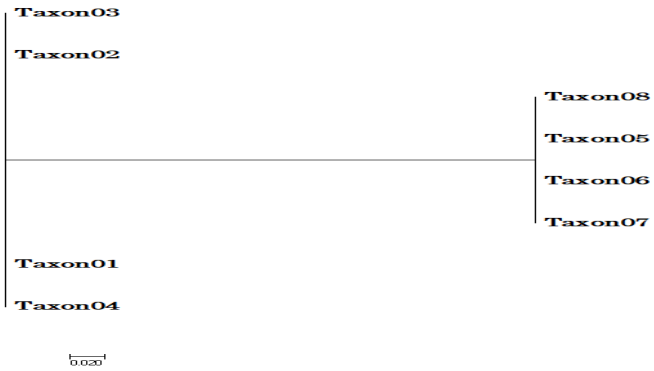


Figure 4: Maximum Likelihood Tree of 8 taxa constructed by RAxML from an alignment of $4,000,000$ columns with R set to 5000

group consists of 4 isolates. The data show a scenario where the species in each group have little variations among themselves (about $100 - 1000$ informative sites in an alignment of 4 or 5 million columns), but any two isolates from different groups have many variations (20% of the total alignment). The generalized time-reversible process (GTR) model [21] was used as the nucleotide model of substitution to generate these nucleotide sequences.

We used alignments of 2 different lengths: 4 and 5 million columns. Let R denote the ratio of the number of informative columns between taxa from different groups to the number of informative columns between two taxa from the same group. For each of the two alignment lengths, we used 4 different values of R: 1000, 2000, 5000 and $10,000$. For each of these 8 data types, 1000 alignments of that type were generated. On each of these alignments, a tree like the one in Figure 4 was constructed by RAxML.

Figure 4 shows two groups of taxa with low resolution.

Our method for selecting taxa with low resolution was applied on this tree. It correctly identified the two groups for all the alignments.

Then our method for finding informative columns from the alignment was used for these closely related taxa. A large percentage of informative columns within those groups were identified by our method. The average number of columns (over 1000 alignments and 2 groups with low resolution for each alignment) identified as informative are listed in Table 3. We know the number of columns generated by Seq-Gen for each of those groups. The percentages indicate the fraction of those columns identified as informative.

Table 3: Number (Percentage) of Informative Columns

Value of R	Length of the Alignment	
	$4,000,000$ columns	$5,000,000$ columns
1000	623 (77%)	730 (73%)
2000	297 (74%)	386 (77%)
5000	127 (79%)	148 (74%)
10000	53 (66%)	77 (77%)

Then the phylogeny of those closely related taxa was constructed from the alignment consisting of only informative columns. For all the alignments, the reconstructed phylogeny correctly grouped $Taxon01$ with $Taxon02$, $Taxon03$ with $Taxon04$, $Taxon05$ with $Taxon06$, and $Taxon07$ with $Taxon08$.

We computed the Robinson-Foulds distance [15] of the trees T_w (constructed from the whole alignment) from the tree in Figure 3. We also computed the same distance for reconstructed trees from informative columns (T_r). The average distances over 1000 alignments for 8 data types are presented in Table 4.

Table 4: Robinson-Foulds Distance of Trees Built from Whole Alignment (T_w) and from Informative Columns (T_r)

Value of R	Length of the Alignment			
	$4,000,000$ columns		$5,000,000$ columns	
	T_w	T_r	T_w	T_r
1000	3.998	0	3.998	0
2000	3.998	0	3.996	0
5000	3.982	0	3.994	0
10000	3.986	0	3.996	0

From Table 4, we can conclude that the trees constructed from the whole alignment have a Robinson-Foulds distance of 4 for most of the alignments. The evolutionary history presented in the phylogeny constructed from the whole alignment does not completely agree with the tree used to generate the alignment. If we identify informative columns for the closely related taxa and reconstruct the phylogeny of those groups using only informative columns, we get the trees with a Robinson-Foulds distance of 0 from the tree used to generate the alignment.

4 Conclusion

Sequence-based methods for constructing phylogeny of many species have been widely used to understand the evolutionary relationships among them. Sometimes one or more specific regions for some species contain more variations than other parts of the genome [8]. Detection of these regions and identification of the species (with variations among themselves in a small part of the genome) is important for constructing the representative history of evolution. We have demonstrated that our method works properly for identifying such clades with low resolution in the tree on both real and simulated data. And our method selected many informative columns for those clades. Our method for finding informative columns does not require repeated construction of maximum likelihood trees from the whole alignment. Due to its linear complexity in the length of the alignment, our method is efficient in finding informative columns from large alignments.

All the instances of simulated data presented in this paper are built from the tree in Figure 3. This tree has a simple structure where each subtree with low resolution contains only 4 isolates. Besides this simple structure, our method is expected to work on a complex branching structure of many subtrees, where the main branches connecting the subtrees have high resolution but each subtree has low resolution. This complex structure is expected to cause difficulty to existing methods based on maximum likelihood and parsimony. We are currently working on generating simulated datasets from such a complex structure and on obtaining results on the performance of our method and existing methods.

In this work, we have assumed that closely related isolates have a single history of evolution, i.e. all informative columns support one history. In future work, we will address a problem in which informative columns support different histories of evolution. Construction of multiple trees for different genome regions from informative columns is useful in revealing the different evolutionary histories of these genome regions.

References

[1] Johanne Ahrenfeldt, Carina Skaarup, Henrik Hasman, Anders Gorm Pedersen, Frank Møller

Aarestrup, and Ole Lund. Bacterial whole genome-based phylogeny: construction of a new benchmarking dataset and assessment of some existing methods. *BMC Genomics*, 18(1):19, Jan 2017.

[2] Avner Bar-Hen, Mahendra Mariadassou, Marie-Anne Poursat, and Philippe Vandenkoornhuyse. Influence function for robust phylogenetic reconstructions. *Molecular biology and evolution*, 25(5):869–873, 2008.

[3] Joseph Felsenstein. Evolutionary trees from dna sequences: a maximum likelihood approach. *Journal of molecular evolution*, 17(6):368–376, 1981.

[4] Joseph Felsenstein and Gary A Churchill. A hidden markov model approach to variation among sites in rate of evolution. *Molecular biology and evolution*, 13(1):93–104, 1996.

[5] Pawel Górecki and Oliver Eulenstein. Refining discordant gene trees. *BMC Bioinformatics*, 15(13):S3, Nov 2014.

[6] Stéphane Guindon, Jean-François Dufayard, Vincent Lefort, Maria Anisimova, Wim Hordijk, and Olivier Gascuel. New algorithms and methods to estimate maximum-likelihood phylogenies: assessing the performance of phyml 3.0. *Systematic biology*, 59(3):307–321, 2010.

[7] Stéphane Guindon and Olivier Gascuel. A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood. *Systematic biology*, 52(5):696–704, 2003.

[8] Xiaoqiu Huang, Anindya Das, Binod B Sahu, Subodh K Srivastava, Leonor F Leandro, Kerry ODonnell, and Madan K Bhattacharyya. Identification of highly variable supernumerary chromosome segments in an asexual pathogen. *PloS one*, 11(6):e0158183, 2016.

[9] Thomas H Jukes, Charles R Cantor, HN Munro, et al. Evolution of protein molecules. *Mammalian protein metabolism*, 3(21):132, 1969.

[10] Mary K Kuhner and Joseph Felsenstein. A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates. *Molecular biology and evolution*, 11(3):459–468, 1994.

[11] Sudhir Kumar, Glen Stecher, and Koichiro Tamura. Mega7: Molecular evolutionary genetics analysis version 7.0 for bigger datasets. *Molecular biology and evolution*, 33(7):1870–1874, 2016.

[12] G William Moore, M Goodman, and J Barnabas. An iterative approach from the standpoint of the additive hypothesis to the dendrogram problem posed by molecular data sets. *Journal of Theoretical Biology*, 38(3):423–457, 1973.

[13] Andrew Rambaut and Nicholas C Grass. Seq-gen: an application for the monte carlo simulation of dna sequence evolution along phylogenetic trees. *Bioinformatics*, 13(3):235–238, 1997.

[14] Vincent Ranwez and Olivier Gascuel. Quartet-based phylogenetic inference: improvements and limits. *Molecular biology and evolution*, 18(6):1103–1116, 2001.

[15] David F Robinson and Leslie R Foulds. Comparison of phylogenetic trees. *Mathematical biosciences*, 53(1-2):131–147, 1981.

[16] Fredrik Ronquist, Maxim Teslenko, Paul Van Der Mark, Daniel L Ayres, Aaron Darling, Sebastian Höhna, Bret Larget, Liang Liu, Marc A Suchard, and John P Huelsenbeck. Mrbayes 3.2: efficient bayesian phylogenetic inference and model choice across a large model space. *Systematic biology*, 61(3):539–542, 2012.

[17] N Saitou and M Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425, 1987.

[18] Xing-Xing Shen, Chris Todd Hittinger, and Antonis Rokas. Contentious relationships in phylogenomic studies can be driven by a handful of genes. *Nature Ecology & Evolution*, 1:0126, 2017.

[19] Alexandros Stamatakis. Raxml version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics*, 30(9):1312–1313, 2014.

[20] David Swofford and Douglas P.. Begle. *PAUP: Phylogenetic Analysis Using Parsimony, Version 3.1, March 1993.* Center for Biodiversity, Illinois Natural History Survey, 1993.

[21] Simon Tavaré. Some probabilistic and statistical problems in the analysis of dna sequences. *Lectures on mathematics in the life sciences*, 17(2):57–86, 1986.

[22] Michael S Waterman and Temple F Smith. On the similarity of dendrograms. *Journal of Theoretical Biology*, 73(4):789–800, 1978.

A Data-driven Biomarker Computational Model for Lung Disease Classification

David Gnabasik and Gita Alaghband
Computer Science and Engineering, University of Colorado Denver
Denver, CO USA
(David.Gnabasik, Gita.Alaghband)@ucdenver.edu

Abstract

We develop a data-driven computational model that reliably classifies individual patient into one of 7 non-overlapping lung disease clinical types within our dataset: healthy non-smokers, smokers diagnosed with and without chronic obstructive pulmonary disease (COPD), adenocarcinoma, squamous cell carcinoma, cystic fibrosis, and acute lung injury. Panels of 12 cytokine blood serum biomarker measurements precisely classify both known and unknown patients into one of these distinct clinical types. Our model classifies clinical types and patients directly from the conditional relationships of noisy, incomplete, and variable protein concentration measurements, including outliers. Biomarker concentration measurements induce discrete state variables through a binning algorithm that exposes the conditional relationships and dependencies among the concentration data. A unique application of an XOR operation on the state space extracts the patterns identifying the set of distinctive features for each clinical type. Our model builds a discrete topological structure from a baseline data set, and is developed using several novel schemes designed specifically for this analysis. The result is a multidimensional space representing a characteristic set of states within each clinical type population.

Keywords: cytokine proteomic biomarkers; computational model; lung disease.

1 Introduction

According to the American Lung Association, an estimated 158,080 Americans are expected to die from lung cancer in 2016 [1]. The 7 lung diseases analyzed here account for some of the most frequent forms of lung disease, with COPD as the fourth leading cause of death in the United States [2]. Respiratory diseases are of multiple origin, and the selected clinical types cover a wide spectrum of suspected causes. More accurate and cost-effective diagnosis is needed so that people with lung diseases are accurately and cost-effectively diagnosed and then treated accordingly, given that Guarascio *et al* declare that not enough is known regarding ideal therapy selection [3].

The use of protein-based biomarkers of lung disease is rapidly advancing, as reviewed by Jun-Chieh *et al* [4], but reliably measuring proteomic biomarker concentrations is difficult due to technical and biological variation, their wide dynamic range of concentrations and numerous post-translational modifications [5]. Despite these variations, we have developed a data-driven Biomarker Computational Model for Lung Disease Classification (BCM-LDC) that reliably distinguishes among various clinically diagnosed lung disease types within our dataset. BCM-LDC hypothesizes that biomarker interactivity induces a distinctive set of host-response protein concentration values for each clinical type, and that certain concentration patterns revealed by these proteins remain characteristically invariant.

BCM-LDC uses a data-driven, supervised-selection learning model; that is, constrained by the limited amount of training data, the model enumerates all possible combinations of biomarker state spaces, then selects that space which most accurately classifies the data into their known clinical types.

In the background §2, we review the suitability of cytokine proteins as host-response biomarkers, the sources of analyzed data, and the difficulties in modeling biological variation given the constraints governing the model, including the issues of overfitting and working within a high-dimensional parameter space. The computational model §3 describes how protein concentrations are topologically modeled and analyzed. §4 presents the experimental results. §5 describes several validation studies, and §6 concludes the paper.

2 Background

2.1 Host-Response Biomarkers

We investigate whether targeted protein variables act as disease state signals due to the existence and modulating strength of their relative and mutual effects upon each other. Our data-driven computational model, BCM-LDC, classifies clinical types and patients directly from the marginal and conditional relationships of biomarker concentration measurements. BCM-LDC selects the unique set of biomarkers – given a small number of biological and statistical assumptions – whose protein host-response topology corresponds to a patient's clinical type. BCM-LDC represents a space of concentration distributions built upon computable discrete states which classifies patients into clinical types, despite significant data variation.

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

Cytokine proteins are secreted by components of the adaptive immune system, and they act as effectors and modulators of lung tissue inflammatory response [6]. The 12 baseline cytokine biomarkers used in this study {EGF, IFNG, IL1A, IL1B, IL2, IL4, IL6, IL8, IL10, MCP1, TNFA, VEGF} (EGF: epidermal growth factor; IFNG: interferon γ; IL: interleukin; MCP: monocyte chemo-attractant protein; TNF: tumor necrosis factor; VEGF: vascular endothelial growth factor) were chosen because of their known sensitivity in host-response to various lung diseases [7], so that concentrations of circulating cytokines in blood serum may be associated with lung disease survival [8].

2.2 Data Sources

BCM-LDC is constructed using host-response cytokine biomarker concentration data from 343 patients given to us in standard units of pico-grams 10^{12} grams per milliliter (pg/ml). Any other data sets obtained from the literature – such as Healthy Serum – are standardized to these units. This baseline data set includes 7 clinical types from which the 12 protein biomarkers are measured. The number of patients per clinical type ranges from 24 to 56 (see Table 4). The **Q=12** baseline biomarkers {EGF, IFNG, IL1A, IL1B, IL2, IL4, IL6, IL8, IL10, MCP1, TNFA, VEGF} measured from each patient's blood serum are chosen because of their known or suspected relationship to lung disease. Two specimens are collected from each patient at the same time, and these two specimens are averaged over each biomarker to provide a single biomarker panel of 12 measurements per patient, except in cases of missing data. Each of the 343 patients are expertly diagnosed as belonging to only one of 7 lung-related clinical types $\boldsymbol{C_t, 1 \leq t \leq 7}$ adenocarcinoma, squamous cell carcinoma, never smokers, smokers with chronic obstructive pulmonary disease (COPD), smokers without COPD, acute lung injury, or cystic fibrosis [9]. We then sequestered a random 10% of these baseline data for subsequent model validation, leaving 310 patients to train to model. There are 659 missing biomarker measurements out of a possible $\boldsymbol{310 * 12 = 3720}$ (82.3% complete) for a total of 3061 measured values. Only 39 of the 343 patients (11.4%) have all 12 biomarker measurements, but 85.4% have 9 or more biomarkers. A total of 17.7% biomarker values are missing from the baseline data set. The mode of the measurements per patient panel is 10. The mean is 9.84. No data was interpolated or averaged to fill in missing data.

Standard protein 2-D gel electrophoresis assay techniques are used to consistently collect homogeneous blood serum specimens. The first five data sets are all from the same unpublished set of experiments [Acknowledgement A] conducted at laboratories at the University of Colorado Health Sciences Center (UCHSC). The last two data sets, cystic fibrosis and acute lung injury, are from different experiments although the wet-lab protocols and analytics are performed in the same way as the first five data sets [Acknowledgement B]. To minimize batch effects, both laboratories incorporated a standard sample in each electrophoresis gel which was subsequently subtracted during analysis, and both used the Cy2 channel from each gel to normalize spot intensities and for automated matching between gels. All patients underwent expert pathology review and have been histologically assigned to one and only one clinical type, provided with the original data sets. The small error bars in Figures 2 and 3 below suggest these data were produced precisely and with quantitative accuracy.

There are many more data values than targeted variables, the 12 biomarkers, which avoids the issue of overfitting. We are working directly with precise concentrations of secreted proteins expressed in blood serum. Even though differences have been uncovered in protein expression between normal and diseased tissues that may have specificity for different tumor types [10], tissue extraction is both costly and invasive. We justify our sampling strategy because it is non-invasive, generates a large set of data with quantitative accuracy involving a small number of targeted variables, and works with a homogeneous composition indicative of the entire organism.

During our initial experiments, we found that any method based upon averaging – such as logistic regression, cosine similarity, or the machine learning Classify function in Wolfram Mathematica© v11.1 – did not classify the baseline clinical types with a sufficient degree of accuracy. Therefore, our subsequent work focused on developing a computational model that processed the entire set of individual concentration values and not just population averages.

3 Computational Model Details

BCM-LDC hypothesizes that interactivity among the biomarkers induces a distinctive concentration distribution as conditioned by the relative concentrations of the other biomarkers. A binning algorithm discretizes the concentration values of every combination of paired biomarkers variables into fixed-sized bins that produces a characteristic multidimensional state space for each clinical type. The binning algorithm is designed to produce both occupied and empty discrete bin states, what we call a *discrete topological structure* (DTS). The bin state pattern that best distinguishes among the clinical type populations is computed by an XOR operation on each possible state space, which also extracts the set of distinctive variable bin states for each clinical type. The distinctive bin state space is then used to assign new patients into one population type given a patient's set of biomarker concentration values. BCM-LDC is briefly presented below.

3.1 Formulating the Computational Model

Our goal is to develop a model that represents the conditional relationships of expressed host-response biomarkers. The first problem is to discretize the biomarker concentration values for every clinical type – paired biomarker combination $\mathbf{CB_r}$, producing a set of bin sizes and number of bins ($\mathbf{W_r}$, $\mathbf{N_r}$, $\mathbf{1 \leq r \leq R}$). A $\mathbf{CB_r}$ is defined as the

aggregate concentration data from each of these pairs of biomarkers within each clinical type. Each clinical type has **77** different combinations of pairs of biomarkers, including pairs of the same biomarker. The binning algorithm bin size computation maximizes the number of occupied bins **Ô** (O-hat), separating concentration data values by the highest possible resolution, while minimizing the number of gaps or empty bins **Õ** (O-tilde) where no data values reside. Empty bins are considered non-permissible data states. BCM-LDC computes a different total number of bins (states) $\mathbf{N_r}$, and bin size $\mathbf{W_r}$, for each combination $\mathbf{CB_r}$. The model computes the probability of each concentration data point belonging to its bin within each $\mathbf{CB_r}$ combination.

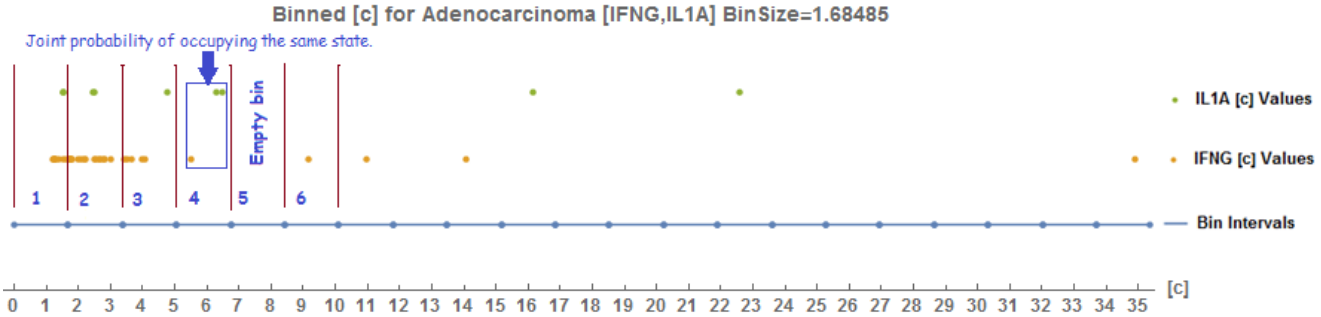


Figure 1: Assigning Adenocarcinoma concentration [c] values to bin states.

3.2 Formulating the Discrete Topological Structure (DTS)

The interactive relationships between each pair of biomarkers {$\mathbf{B_1}$, $\mathbf{B_2}$} are represented by three types of probability. The model computes the pair's *joint occurrence* matrix $\mathbf{M_{C\text{-}joint}}$ – the probability that biomarker $\mathbf{B_2}$ measured at concentration $\mathbf{[c_2]}$ occurs at the same time biomarker $\mathbf{B_1}$ is measured at concentration $\mathbf{[c_1]}$. The model also computes their *conditional probabilities* where, given concentration measurement $\mathbf{[c_1]}$ for $\mathbf{B_1}$, how likely is the measured concentration $\mathbf{[c_2]}$ for $\mathbf{B_2}$. Call this matrix $\mathbf{M_\beta}$. The model uses *marginal probabilities* to represent the influence of individual biomarkers – the probabilities of various concentration values of a subset of biomarker variables without reference to the values of the other variables being considered. Call this matrix $\mathbf{M_\alpha}$. These three types of computed probability taken together express the mutual interactivity and distribution of the biomarker concentration measurements to reveal concentration patterns characteristic of each clinical type. We equate these probability concepts to a *discrete topological structure (DTS)* matrix with equation 1. A data-driven DTS matrix is computed for each $\mathbf{CB_r}$. and the matrix (i.e., the specific set of paired biomarkers) that produces the most accurate set of patient classifications per clinical type is designated $\mathbf{M_C}$ for that population.

$$\mathbf{M_C} = \mathbf{M_{C-joint}}(\mathbf{1} - \mathbf{M_\alpha}) + \mathbf{M_\beta} \qquad (1)$$

In equation 1, $\mathbf{M_{C\text{-}joint}}$ is the population joint occurrence matrix, **1** is a complete matrix of ones (not the identity matrix), $\mathbf{M_\alpha}$ is the **α** interaction matrix of marginal probabilities, and $\mathbf{M_\beta}$ is the **β** interaction matrix of conditional probabilities for the clinical type. The DTS equation is implemented in terms of matrices of conditional and marginal probabilities involving bivariate pairs of biomarkers, each of which are indexed by their respective set of discrete bin states as computed by the binning algorithm. Pseudo-code for the binning algorithm is given in Algorithm 1 below, where $\mathbf{D_r}$ refers to as the combined set of observed concentration data values within each $\mathbf{CB_r}$, for a specific clinical type and biomarker pair {$\mathbf{B_i}$, $\mathbf{B_j}$}.

Algorithm 1: Pseudo-code for the Bin-Min-Max algorithm.

Inputs: D_r: set of concentration data for given CB_r ; maxNbins: max number of bins. **Outputs**: returns W_r, N_r

```
1. foreach combination Dr = { D(Bi), D(Bj) }
2. # Initialize number of bins (Nr), bin step
    size (binInc), bin size (Wr), number of
    empty bins (emptyBins), tmp = 0.
3. Nr ← binInc ← ∜maxNbins
4. Wr ← |max(Dr) - min(Dr)| / Nr
5. emptyBins ← Count_Empty_Bins(Dr, Nr, Wr)
6. result ← |Wr - log_e(emptyBins)|
7. while (result < tmp and Nr < maxNbins -
    binInc) do
8.   Nr ← Nr + binInc
9.   Wr ← |Max(Dr) - Min(Dr)| / Nr
10.  emptyBins ← Count_Empty_Bins(Dr, Nr, Wr)
11.  tmp ← result
12. If (emptyBins > 0) result ← |Wr -
    log_e(emptyBins)|) else result ← 1
13. end while
14. Return (Wr, Nr)
15. end foreach
```

The output of the Max-Bins-Min-Empty-Bins binning algorithm is a bin size $\mathbf{W_r}$ and the number of bins $\mathbf{N_r}$ for each clinical type – paired biomarker combination $\mathbf{CB_r}$. Each value in a set of combined concentration values is assigned to a single bin, but multiple concentration values can be assigned to the same bin, as plotted in Figure 1 for Adenocarcinoma biomarkers {$\mathbf{B_i}$ = IFNG, $\mathbf{B_j}$ = IL1A}. The top 2 **[c]** rows in Figure 1 refer to their actual concentration values measured in pg/ml. These **[c]**values are mapped to specific bin states in the Bin Intervals row. Many of the **[c]** values are grouped in the first few bins. The first 6 states are labeled numerically, and bin 5 is the first empty bin out of

the 23 bins. Bins 1 through 4 illustrate the joint probabilities of IL1A and IFNG values occupying the same state. Additional details for computing each DTS matrix are given in the next section.

3.3 Computing the DTS Matrix $\mathbf{M_C}$

BCM-LDC computes the population joint probabilities for $\mathbf{D_r}$ for each clinical type $\mathbf{C_t}$, combination $\mathbf{CB_r} \in \mathbf{C_t}$, biomarker $\mathbf{B_i} \in \mathbf{CB_r}$, bin $\mathbf{b}$ from 1 to $\mathbf{N_r}$ using equation 2, where $\mathbf{G_b}$ is the number of **[c]** values of $\mathbf{B_i}$ in bin $\mathbf{b}$. The result $\mathbf{P_i}$ is the vector of probabilities for observing the biomarker concentrations in each bin, oftentimes zero. A bin probability equals the number of concentration values $\mathbf{G_b}$ grouped in each bin divided by $|\mathbf{D_r}|$ so that the sum of probabilities over the set of bins is 1.

$$\mathbf{P}_i[\mathbf{b}] = \frac{\mathbf{G_b}}{|\mathbf{D_r}|}, \mathbf{1} \le \mathbf{r} \le \mathbf{R}, \mathbf{1} \le \mathbf{b} \le \mathbf{N_r} \qquad (2)$$

The population joint occurrence matrix $\mathbf{M_{C\text{-}joint}}$ is computed by multiplying each bin probability $\mathbf{P_i}$ for biomarker $\mathbf{B_i}$ with each bin probability $\mathbf{P_j}$ for biomarker $\mathbf{B_j}$, where $\mathbf{B_i}$ is indexed by $\mathbf{i}$ from 1 to the number of bins $\mathbf{N_{Bi}}$ for biomarker $\mathbf{B_i}$ and $\mathbf{B_j}$ is indexed by $\mathbf{j}$ from 1 to the number of bins $\mathbf{N_{Bj}}$ for biomarker $\mathbf{B_j}$. Equation 3 multiplies two vectors (one row vector and one column transposed) together element-wise as an outer product to form a 2-dimensional matrix for that biomarker combination of $\mathbf{B_i}$ and $\mathbf{B_j}$. The dimensions of $\mathbf{M_{C\text{-}joint}}$, one for each $\mathbf{CB_r}$, is $\mathbf{N_{Bi}}$ x $\mathbf{N_{Bj}}$. Bins 1 through 4 in Figure 1 illustrate joint occurrence values greater than zero.

$$\mathbf{M_{C-joint}}(\mathbf{i}, \mathbf{j}) = \mathbf{P}(\mathbf{B_i}) \otimes \mathbf{P}(\mathbf{B_j})^{\mathbf{T}} \qquad (3)$$

The population marginal distributions $\mathbf{M_{i\text{-}marg}}$ and $\mathbf{M_{j\text{-}marg}}$. are computed by equations 4 and 5.

$$\mathbf{M_{i-marg}}(\mathbf{i}) = \sum_{\mathbf{j=1}}^{\mathbf{NBi}} \mathbf{M_{C-joint}}(\mathbf{i}, \mathbf{j}), \mathbf{1} \le \mathbf{i} \le \mathbf{N_{Bi}} \qquad (4)$$

$$\mathbf{M_{j-marg}}(\mathbf{j}) = \sum_{\mathbf{i=1}}^{\mathbf{NBj}} \mathbf{M_{C-joint}}(\mathbf{i}, \mathbf{j}), \mathbf{1} \le \mathbf{j} \le \mathbf{N_{Bj}} \qquad (5)$$

The $\boldsymbol{\alpha}$ interaction matrix $\mathbf{M_\alpha}$ – the matrix from equation 1 with dimensions $\mathbf{N_{Bi}}$ x $\mathbf{N_{Bj}}$ – is composed as the transposition of $\mathbf{M_{i\text{-}marg}}$ repeated $\mathbf{N_{Bj}}$ times. The population conditional probability matrix $\mathbf{M_{C\text{-}cond}}$ for a pair of biomarkers $\{\mathbf{B_i}, \mathbf{B_j}\}$ – one per $\mathbf{CB_r}$ – is computed as an element-by-element matrix division in equation 6.

$$\mathbf{M_{Ci-cond}} = \mathbf{M_{C-joint}}/\mathbf{M_{j-marg}}$$
$$\mathbf{M_{Cj-cond}} = \mathbf{M_{C-joint}}/\mathbf{M_{i-marg}} \qquad (6)$$

The $\boldsymbol{\beta}$ interaction matrix $\mathbf{M_\beta}$ is defined in equation 7 as $\mathbf{P_i}$ divided element-wise by $\mathbf{P_j}$ (from equation 2).

$$\mathbf{M_\beta}(\mathbf{i}, \mathbf{j}) = \mathbf{P_i}/\mathbf{P_j}, \text{given } \mathbf{P_j} > 0, \text{else } 0 \qquad (7)$$

Equation 1, derived from equations 2–7, computes a DTS matrix $\mathbf{M_C}$ for each $\mathbf{CB_r}$ that represents the conditional probability relationship between all pairs of biomarkers within each population. Each $\mathbf{CB_r}$ combination has a characteristic vector of occupied bin states $\hat{\mathbf{O}}$ and empty bin states $\tilde{\mathbf{O}}$ out of a possible number of bins $\mathbf{N_r}$ as calculated by the binning algorithm. Each $\mathbf{CB_r}$ combination now composes an object with the following properties, which will be used to find out the set of distinguishing biomarkers per clinical type:

- clinical type population $\mathbf{C_t}$,
- biomarker pair $\{\mathbf{B_i}, \mathbf{B_j}\}$,
- bin size $\mathbf{W_r}$,
- number of bins $\mathbf{N_r}$,
- bin state vector $[\mathbf{P_i}, \hat{\mathbf{O}}, \tilde{\mathbf{O}}, \mathbf{G_b}]$,
- set of observed concentration values $\mathbf{D_r}$,
- matrices $\mathbf{M_C}$, $\mathbf{M_{C\text{-}cond}}$, $\mathbf{M_{C\text{-}joint}}$, $\mathbf{M_{C\text{-}marg}}$, $\mathbf{M_\alpha}$, $\mathbf{M_\beta}$.

The main advantage of using calculated DTS values instead of raw concentration [c] values is the normalization of scale. Figure 2 plots all biomarker concentration measurements for clinical type Adenocarcinoma, covering a wide range of scales. Figure 3 plots the corresponding Adenocarcinoma DTS values. The binning algorithm calculates the DTS values to all lie within one order of magnitude for every clinical type, and the DTS values are more regularly spaced.

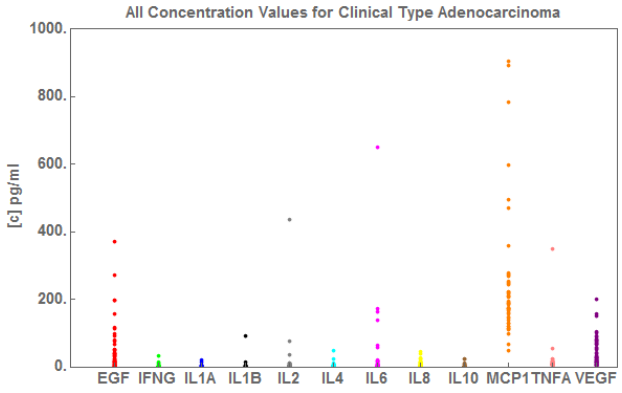


Figure 2: All 12 biomarker Adenocarcinoma [c] values.

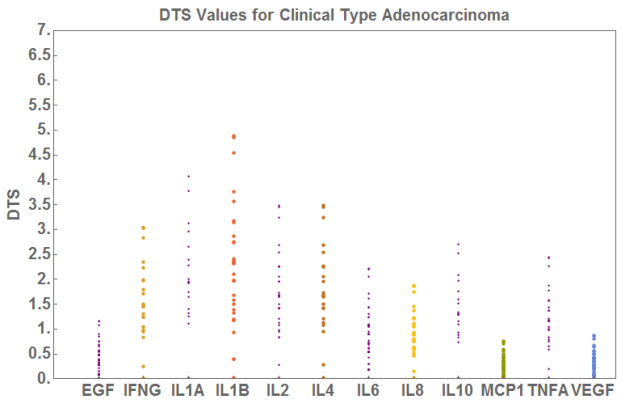


Figure 3: All 12 biomarker Adenocarcinoma DTS values.

3.4 Distinguishing Biomarkers

To reveal the distinguishing biomarkers for each clinical type, BCM-LDC forms a coordinate system of the bin state probability values and the DTS values per biomarker instead of comparing concentration values. The bin states are transformed to matrix form to expose their characteristic and distinguishing states. These integer matrices are constructed by first standardizing the bin state probability and DTS values. The probability values are multiplied by 100 and rounded to integers as percent values along the x-axis to form a standard 100 cells. The corresponding DTS values are raised as exponents to the natural logarithm and rounded to integers, standardizing the y-axis to 256 cells, and starting from the upper left corner. This forms a cellular structure where a whole integer in a cell indicates the presence of a probability–DTS value and 0 otherwise. An element-by-element **XOR** operation between the cellular structures of any two clinical types of the same biomarker reveals which clinical type probability–DTS bin values are unique between those two clinical types. An elaboration of this logic obtains the complete list of distinguishing bin states of the same biomarker among all clinical types. The objective is the same – to identify those matrix cells that are occupied by one and only one clinical type for that biomarker, as described next.

BCM-LDC replaces the occupied matrix integer values with unique $\mathbf{2^n}$ clinical type identifiers (e.g., Adenocarcinoma: 2^1=2), and then adds every matrix together per biomarker so that each matrix cell contains zero, one, or more than one clinical type identifier. An element-by-element $\mathbf{log_2}$ operation that returns a whole integer identifies a single clinical type occupying that cell. This method depends upon the fact that a binomial coefficient (m choose n) (mod 2) is computable using an $_n\mathbf{XOR}_m$ operation. Figure 4 plots the integer matrix for biomarker IFNG for all clinical types, where Adenocarcinoma is distinguished by 3 (red) circled cells. The (blue) circled value of **34**=2+32 indicates that both Adenocarcinoma and Smokers without COPD (2^5=32) exist in the same cell.

0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
64.	64.	64.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	2.	32.	16.	4.	2.	34.	0.	32.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	8.	8.	0.	8.	0.	0.	8.	0.	0.	0.
0.	0.	0.	0.	0.	0.	2	0.	0.	0.	0.	0.
0.	0.	32.	64.	0.	0.	0.	0.	16.	0.	16.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	8.	8.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	32.	0.	16.	0.	0.	0.	0.	0.
0.	64.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	64.	0.	0.	64.	0.
0.	0.	32.	0.	0.	0.	0.	16.	0.	0.	0.	0.

Figure 4: Partial integer matrix for biomarker IFNG for all 7 clinical types.

The 12 individual integer matrices produced for each clinical type can be consolidated into 3 dimensions to plot their distinguishing biomarkers with respect to the aforementioned probability cell and DTS cell states. Figure 5 plots the distinguishing probability cell and DTS cell states of all the clinical types together. We observe that the range of probability values is low in the Probability dimension – no single biomarker overwhelms any of the others in terms of frequency. It is also clear that the DTS coordinate effectively separates out the clinical types. Interestingly, Never Smokers (blue) displays the most variation among all the clinical types – one is "normal" in a wide variety of states.

4 Experimental Results

Table I lists the common distinguishing biomarkers per clinical type over the 10-fold cross-validation study (see §5.1).

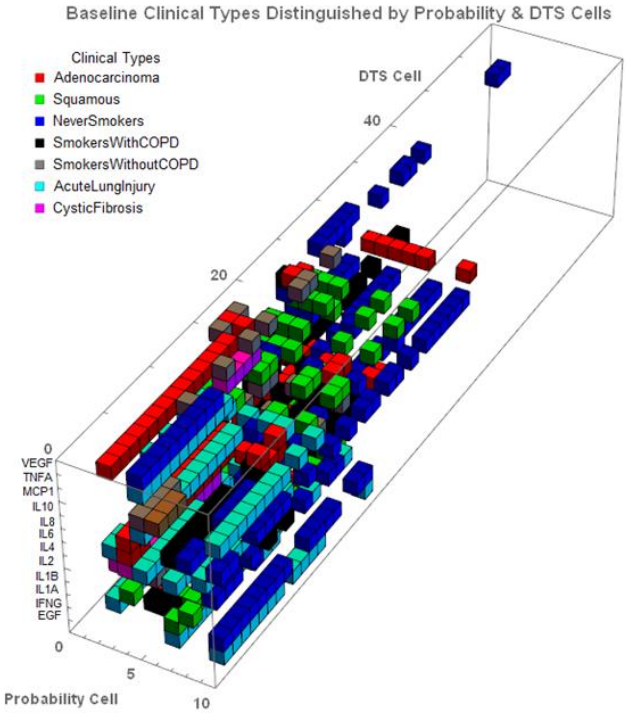


Figure 5: Clinical types distinguished by Probability and DTS states.

TABLE I. DISTINGUISHING BIOMARKERS PER CLINICAL TYPE IN THE PROBABILITY – DTS DIMENSIONS.

Clinical Type Classification C_t	**N Distinguishing Biomarkers**	**Patient Counts**	**Total Bins**	A_t
Adenocarcinoma	**6:** IL1B IL4 IL6 IL8 MCP1 VEGF	53	444	0
Squamous	**6:** IL1B IL2 IL8 IL10 MCP1 TNFA	44	1664	0
Never Smokers	**4:** EGF IFNG TNFA VEGF	55	624	3
Smokers with COPD	**4:** EGF MCP1 TNFA VEGF	49	492	0
Smokers without COPD	**2:** EGF VEGF	53	386	0
Acute Lung Injury	**12:** EGF IFNG IL1A IL1B IL2 IL4 IL6 IL8 IL10 MCP1 TNFA VEGF	62	572	0
Cystic Fibrosis	**1:** IL1A	27	360	0

5 Validation Studies

We can now assign an unknown patient sample **z** to a known clinical type by computing the patient's DTS matrix $\mathbf{M_z}$ and comparing it to every $\mathbf{M_{Ct}}$. Comparing $\mathbf{M_z}$ to every $\mathbf{M_{Ct}}$ uses a fitness function (equation 8) that decides which clinical type is closest to the unknown sample state.

$$\mathbf{s_z} \in \mathbf{C_t} = \mathbf{min}\left(\mathbf{abs}(\mathbf{M_Z} - \mathbf{M_{Ct}})\right), \mathbf{1} \leq \mathbf{t} \leq \mathbf{C_T}|\mathbf{B_j}, \mathbf{N_x} \qquad (8)$$

Assigning a unique bin number and bin probability for each sample biomarker value simply involves looking up the corresponding bin number in the known population probability list for that biomarker. The probability of a sample's concentration value is the expected probability of its assigned bin.

5.1 10-Fold Cross-Validation

We conducted a 10-fold cross-validation study on the 343 baseline patients, where 10% of the samples were randomly extracted 10 times using SQL Server's *NewID* function and then running BCM-LDC over each of the different data partitions. Those distinguishing biomarkers that were present in every one of the 10 runs per clinical type are listed in Table I, column 2. The total number of incorrect baseline patient assignments $\mathbf{A_t}$ over the 10 runs is given in the column 5. Three Never Smokers baseline patients over the 10 runs were incorrectly assigned, of which 2 were the same sample. We account for these incorrect assignments by the large variation present in the Never Smokers patients (see the last part of §3.4) and not by missing biomarker values.

During the same 10-fold cross-validation, each of the 10% sequestered (33) patients were correctly assigned to their respective clinical types with the exception of one (the same) Cystic Fibrosis patient assigned as Acute Lung Injury twice. We account for this incorrect assignment by the tiny sample size of the sequestered Cystic Fibrosis patients, which was the smallest to begin with.

5.2 Healthy Serum Validation

Whereas the baseline clinical types were collected by standard 2-D PAGE gel electrophoresis protocols, measurements from 144 "Healthy Serum" serum samples were taken from a different sampling protocol and experimental design (Luminex® fluorescent bead-based immunoassay [11]). Data was not collected for the EGF or IL2 biomarkers, but included the other 10 biomarkers. When processed along with the baseline data sets, all samples were correctly assigned to their Healthy Serum clinical type.

6 Conclusions

We have developed a computational model, BCM-LDC, that reliably distinguishes among 7 given lung pathologies by assigning biomarker concentration values to discrete states despite significant data variation and technical challenges. BCM-LDC distinguishes the set of biomarker variables that uniquely characterize the clinical types under analysis. The source data – concentration values of host-response serum cytokines – serve as adequate biomarker variables. Excluding Cystic Fibrosis and Smokers without COPD, there is no single biomarker pair that distinguishes among all clinical types, though EGF~VEGF does for 4 types. The minimal biomarker pairs that distinguish among the remaining 5 clinical types are {EGF~TNFA or EGF~VEGF or TNFA~VEGF} and {IL1B~IL8 or IL1B~MCP1 or IL8~MCP1}. Whereas the distinguishing biomarkers extracted are data-driven, patient samples are classified into their single clinical type with reliability greater than 99%.

The Discrete Topological Structure computational model distinguishes among the clinical type populations by discretizing concentrations values to populate only certain bin states. The resulting DTS model simplifies the high-dimensional biomarker concentration space so that some distinguishing features of the lung disease space are revealed.

ACKNOWLEDGMENTS

A. We thank Dr. M. Duncan of USHSC for providing us with 5 / 7 original unpublished data sets.

B. We thank Dr. Paul Bunn of USHSC for providing us with 2 / 7 original unpublished data sets.

REFERENCES

[1] Lung Cancer Fact Sheet from the American Lung Association. Available at http://www.lung.org/lung-health-and-diseases/lung-disease-lookup/lung-cancer/resource-library/lung-cancer-fact-sheet.html (Nov. 2016)

[2] P.T. Reid, J.A. Innes. "Respiratory disease", In: Walker BR, Colledge NR, Ralston SH, Penman ID, eds. Davidson's Principles and Practice of Medicine. 22nd ed. Philadelphia, PA: Elsevier Churchill Livingstone; chap 19. (2014)

[3] A.J. Guarascio, S.M. Ray, C.K. Finch, T.H. Self. "The clinical and economic burden of chronic obstructive pulmonary disease in the USA", ClinicoEconomics and Outcomes Research. Jun 17;5:235-45. (2013)

[4] J.T. Jun-Chieh, C. DeCotiis, A.K. Greenberg, W.N. Rom, "Current Readings: Blood-Based Biomarkers for Lung Cancer", Semin Thorac Cardiovasc Surg. (2013) Winter; 25(4): 328–334.

[5] K. Chandramouli, P-Y Qian, "Proteomics: Challenges, Techniques and Possibilities to Overcome Biological Sample Complexity",Human Genomics and Proteomics, vol 2009, 239204.

[6] R. Sivangala, G. Sumanlatha. "Cytokines that Mediate and Regulate Immune Responses", Austin Publishing Group (2015). Innovative Immunology. Available: www.austinpublishinggroup.com/ebooks

[7] L. Enewold *et al*, "Serum concentrations of cytokines and lung cancer survival in African Americans and Caucasians", Cancer Epidemiol Biomarkers Prev. 2009 Jan;18(1):215-22.

[8] C. A. Dinarello, "Proinflammatory Cytokines", Chest 2000;118;503-508.

[9] J. Subramanian, R. Govindan, "Lung Cancer in Never Smokers: A Review", Journal of Clinical Oncology 25 (5): 561–70. (2007)

[10] M.R. Mehan, D. Ayers *et al*, "Protein Signature of Lung Cancer Tissues", PLoS ONE7(4): e35157. (2012)

[11] Biancotto A, Wank A, Perl S, Cook W, Olnes MJ, *et al.* "Baseline Levels and Temporal Stability of 27 Multiplexed Serum Cytokine Concentrations in Healthy Subjects", PLoS ONE 8(12): e76091. doi:10.1371/journal.pone.0076091 (2013)

A Multiscale Model Explains the Circadian Phase Dependent Firing Pattern Variations in Suprachiasmatic Nuclei and the Occurrence of Stochastic Resonance

Shiju S and K Sriram
Center for Computational Biology, Indraprastha Institute of Information Technolgy-Delhi
NewDelhi,110020, India
(shijus, sriramk)@iiitd.ac.in

Abstract

We perform multiscale model simulations to study the role of slow varying mammalian circadian oscillations (in hrs) and Gaussian noise in modulating the rapidly varying firing patterns (in ms) exhibited by the ionic channels of suprachiasmatic nuclei under DD (constant darkness) conditions. Hodgkin-Huxley model near subcritical Hopf bifurcation exhibits noise-induced firing patterns and these patterns, modulated by slow varying circadian gene regulatory network, are highly circadian phase dependent. The simulated firing patterns are also very close to the experimentally observed patterns with a firing rate of 3-10 HZ during subjective day and 0-3 HZ during subjective night. Further, for certain noise intensity, the model's response is maximal, a characteristic feature of stochastic resonance, but surprisingly, we observe it only at certain circadian phases. This is the first instance where it is shown that the slow-varying gene regulatory circadian oscillation along with noise modulates the firing patterns of fast varying voltage gated channel as observed in experiments and exhibits stochastic resonance only in certain circadian phases.

Keywords: Stochastic resonance, multiscale model, circadian firing patterns, signal to noise ratio

1 Introduction

In mammals, Suprachiasmatic nuclei (SCN) is the master oscillator with ≈20000 neurons that exhibit endogenous oscillations with a period close to 24h in gene expression under constant DD conditions [1]. On the other hand, SCN neurons also fire spontaneously due to the opening and closing of ionic channels but with varying frequencies during the 24h cycle. It is well known that electrophysiological properties of neurons are controlled by ionic channels, which in turn depends on the conductance of ionic current. The mean value of conductance varies in a circadian manner [2]. During subjective day, firing frequencies in the range between 3-10 Hz reaches the maximum, while during subjective night, the firing frequency reaches a nadir that ranges between 0-3 Hz [3]. Presently, the mechanism responsible for modulation of firing pattern variations during the 24h cycle is not known and importantly, the bonafide circadian genes like *per1/2*, *Bmal1*, *Cry1/2*, *Rev-Erb*α responsible for interacting with the ionic channels in SCN is not fully elucidated. However, recently, Jones et al. [3] reported that the gene *per1* plays an important role in modulating the firing rate rhythm in SCN neuron. Therefore, we intend to examine the following questions through multiscale model simulations; (i) How slow varying gene regulatory network (GRN) of circadian rhythm of 24 h modulate the fast varying firing patterns (0-10 HZ) of voltage gated channels in SCN (ii) What is the role of noise in modulating the firing patterns in SCN and (iii) How the interplay of noise, slow varying GRN and fast varying ionic channels contribute to different firing patterns during the subjective day and night in circadian systems.

In this work, an attempt has been made to provide plausible explanation for the above questions by building a multiscale model and interestingly, we also study the role of noise in providing optimal firing response in a circadian phase-dependent manner by invoking the concept of stochastic resonance (SR), a paradigm concept in a noise-induced phenomena wherein the presence of noise enhances the quality and detection of weak signals and has a wide range of application particularly in neuronal system. The response of a nonlinear system to noise reaches a maximum for an optimum value of noise intensity is the typical characteristic feature of stochastic resonance. Benzi et al. [4] first showed the SR phenomenon in a dynamical system subjected to both periodic and random perturbation, however, noisy nonlinear systems can display SR even in the absence of external forcing signal [5, 6], called coherence resonance (CR).

Further, we also show through in-silico multiscale model simulations that for the choice of optimal noise intensity, the simulated firing patterns in SCN closely follow the experimental results of Jones et al. [3] and importantly, for a particular circadian time exhibits stochastic resonance. Interestingly, this is the first instance wherein the interplay of different natural signals with two varying time scales emanating from a single neuron in the presence of noise that arises due to random opening and closing of voltage

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

gated channels are used to explain the modulation of firing patterns.

2 Methodology

2.1 Two models with different timescales

We choose two models that have disparate time scales in SCN: (i) a variant of Hodgkin-Huxley (HH) model built by Diekman et al. [7] specifically for SCN neurons and we use this model to capture the dynamics of firing rates in SCN (Figure 1). This model has a timescale in the order of milliseconds (ms) and this model is a slow-fast system with voltage having a slow timescale while the gating variables have a fast timescale. (ii) Gene regulatory model of Goodwin type captures the dynamics of mRNA and proteins and has a time scale in the order of hours. We couple both these models of disparate timescale unidirectionally with *per1* mRNA regulating specific ion channels in the HH model. This is the forcing term of our model. We also add delta-correlated Gaussian noise to the HH model and determine the noise intensity for which SNR is maximum. We describe below in detail the two models, simulations, and validation with the experimental data.

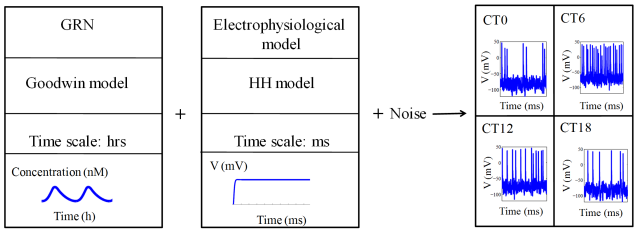


Figure 1: Schema of multiscale SCN model. GRN is based on the Goodwin oscillator model, where the clock concentration varies in circadian manner with a period close to 24 h. The slow varying GRN (in the left column) drives the fast varying electrophysiological HH model, which is below the threshold and in steady state (in the middle column), generates action potential in the presence of noise with a period close to millisecond scale (in the last right column). Note that the firing pattern is highly circadian phase dependent and it is shown here for CT0, 6, 12 and 18 hrs.

2.2 Electrophysiological model of SCN

The current balance equation of SCN by Diekman et al. [7] is provided below, and to this we add Gaussian noise.

$$C\frac{dV}{dt} = I_{app} - I_{Na} - I_{CaL} - I_{CaNonL} - I_{KCa} - I_{K-leak} - I_{Na-leak} + D * w \quad (1)$$

where V and C are the membrane potential and capacitance respectively, and I_{app}, I_{Na}, I_{CaL}, I_{CaNonL}, I_{KCa}, I_{K-leak}, and $I_{Na-leak}$ are the external, sodium, leakage calcium, non-leakage calcium, calcium activated potassium, leakage potassium, and leakage sodium currents respectively. Here w is the Gaussian white noise with zero mean and unit variance. D is the intensity of noise, which has the dimension of current. Here we assume that w represents the combined stochastic activity of the ion channels on the voltage dynamics of the SCN neuron. Remaining equations are same as that appeared in the original model [7]. Parameter values used for simulations are also the same as that of in the original model [7] except that $g_{Na-leak} = 0.052nS$, $g_{KCa} = 180nS$, $g_{K-leak} = 0.15nS$ and $\tau_s = 0.01ms$.

2.3 GRN model of SCN

The GRN model is a Goodwin type oscillator [8] consists of the dynamical variables *per1* mRNA (M_P), PER1 protein (P) and phosphorylated PER1 protein (P_p). The model describes the production of mRNA and protein, their degradation and importantly, to oscillate a delayed-negative feedback of phosphorylated protein is integrated to describe the negative regulation of transcription rate and it is given by Hills equation. These equations are highly nonlinear in nature. The full GRN model is given as follows:

$$\frac{dM_P}{dt} = a\left(\frac{0.001^{n_c}}{0.001^{n_c} + P_p^{n_c}} - M_P\right) \quad (2)$$

$$\frac{dP}{dt} = a(M_P - P) \quad (3)$$

$$\frac{dP_p}{dt} = a(P - P_p) \quad (4)$$

There are two parameters in the above coupled set of nonlinear equations; the scaling parameter a (= 4.46E-8) and the Hills coefficient n_c (= 9). The model exhibits limit cycle oscillations with a period of 23.6 h and this is taken to be the free-running period of mammalian SCN neuron [1] for all the simulations.

3 Simulation results

3.1 Coherence resonance in HH model of SCN without periodic forcing by GRN

Codimension-1 bifurcation diagram for the membrane potential V as a function of $g_{Na-leak}$ is conducted in the absence of noise ($D = 0$, Figure 2). As $g_{Na-leak}$ increases the stable steady state becomes unstable through subcritical Hopf bifurcation, where the stable steady state is surrounded by the unstable limit cycle, which in turn is surrounded by a stable limit cycle. Further increase in $g_{Na-leak}$ leads to a loss of periodic orbits through supercritical Hopf bifurcation that

has unstable steady state surrounded by stable limit cycle. To simulate the noise induced firing of action potential in our model, we choose a subthreshold value of $g_{Na-leak} = 0.052$ that exhibits only stable steady state. Addition of noise provides a transition of the system from steady state to limit cycle oscillations randomly thus producing the train of action potential. The maxima between two spikes is the interspike interval (ISI) and the histogram of the ISI's for all the spikes gives rise to Interspike interval histogram (ISIH). This is done for various noise intensities D, numerical integration has been carried out in Xppaut [9] using stochastic Euler's method [10] with the integration time step $\Delta t = 0.02ms$ and the resulting ISI histograms are shown in Figure 3A. For a low value of D, the time taken by the membrane potential to cross the activation threshold of firing is large and hence the number of spike around the mean interspike interval (ISI) is small. With the increase in the noise intensity, membrane potential quickly crosses the activation threshold and as a consequence, the frequency of the ISI peak increases.

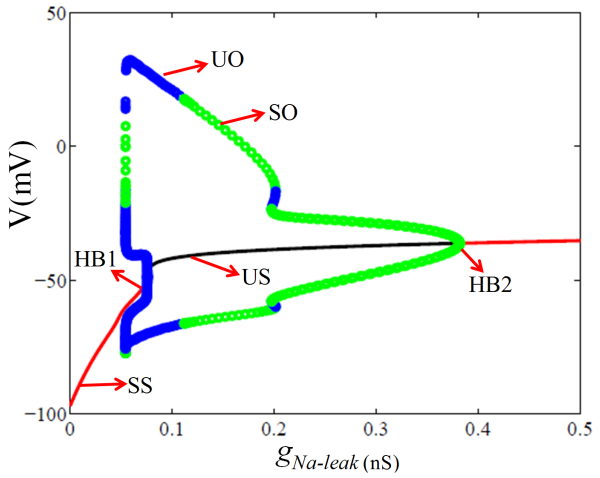


Figure 2: Codimension-1 bifurcation diagram of HH model with $g_{Na-leak}$ as the bifurcation parameter. For lower values of $g_{Na-leak}$ the system is in the stable steady state (Red lines, SS). As $g_{Na-leak}$ increases oscillations appears via subcritical Hopf bifurcation (HB1). Black lines are the unstable steady state (US), blue circles are unstable oscillation amplitude (UO), and green circles are stable oscillations amplitude (SO). Sustained oscillations disappeared via supercritical Hopf bifurcation (HB2) and the system enters the stable steady state. Xppaut [9] was used for simulating the bifurcation diagram.

To measure the system's response to noise, power spectral density (PSD) of the membrane potential, V was computed by averaging out 25 runs of the firing patterns of membrane potential with a duration of 5 min using fast Fourier transform. The averaged PSD is further smoothed by applying Savitzky-Golay filtering method with 100 number of points using MATLAB®(*'sgolay'*). The height of the peak in the power spectra is very small for lower values of noise intensity (Figure 3 B, $D = 1$, black curve). With the increase of noise intensity height of the power spectral peak increases and starts to saturate. To characterize the system response, we compute the measure of signal to noise ratio (SNR) defined in [5] as

$$\beta_s = HQ_s, \quad Q_s = \frac{\omega_p}{\Delta\omega} \tag{5}$$

where H is the height of the peak, ω_p is the frequency at which peak occurs and $\Delta\omega$ is the width of the peak at half maximum height. SNR for different noise intensity is shown Figure 3C, coherence resonance occurs at optimum noise intensity ($D = 2$), where system response is maximum and further increase in noise, SNR starts to decrease. Even though there is a shift in the peaking frequency in power spectra, frequency variation in firing pattern is very less. Hence the coherence resonance is not sufficient to induce experimentally observed circadian variation of firing rate at SCN.

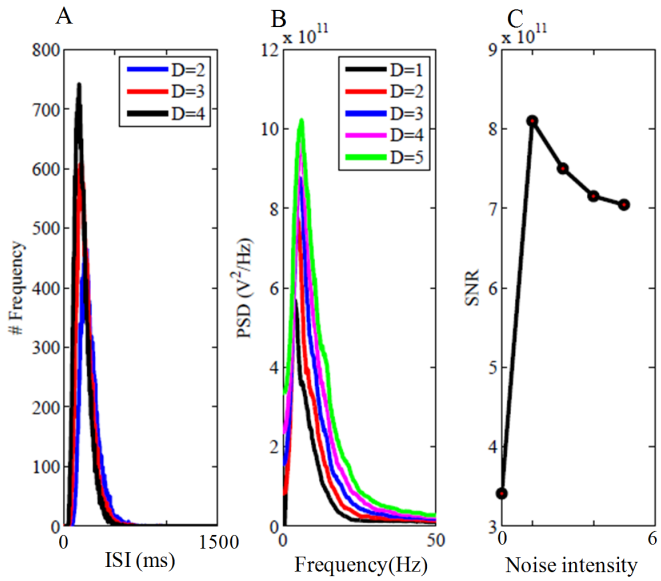


Figure 3: ISIH, PSD, and SNR for HH model without forcing. (A) ISIs are not randomly distributed, most firing periods are near the intrinsic periods of the model ($\approx 200ms$) but the number of spike increases with increase in the noise intensity. (B) Averaged power spectra of membrane potential for different noise intensity. Height of the spectra increases with increase in D and saturated for higher values. Smooth PSD curve is obtained by applying Savitzky-Golay filtering method with 100 number of points. (C) The SNR is calculated from power spectra using the equation 5. SNR reaches a maximum then decreases, characteristics of stochastic resonance.

3.2 Role of slow frequency circadian GRN in regulating firing frequency variations: Multiscale Coupled oscillator model

Now we introduce the slow forcing signal of the gene regulatory network, namely the clock variable M_p that regulates the HH model through the conductance of sodium leak ($g_{Na-leak}$) variable as follows:

$$g_{Na_{leak}} = g_{Na_{leak_{pp}}} M_{P_{Normalized}} + g_{Na_{leak_{base}}}$$

where $g_{Na_{leak_{pp}}} = 0.032nS$, $g_{Na_{leak_{base}}} = 0.02nS$, and $M_{P_{Normalized}}$ is the normalized value of M_P ($M_{P_{Normalized}} = \frac{M_P - M_{P_{min}}}{M_{P_{max}} - M_{P_{min}}}$). The value of $g_{Na-leak}$ is varied in near the subcritical Hopf bifurcation. This circadian coupling with the conductance generates a subthreshold circadian rhythmicity on membrane potential in the absence of noise ($D = 0$) as shown in Figure 4A.

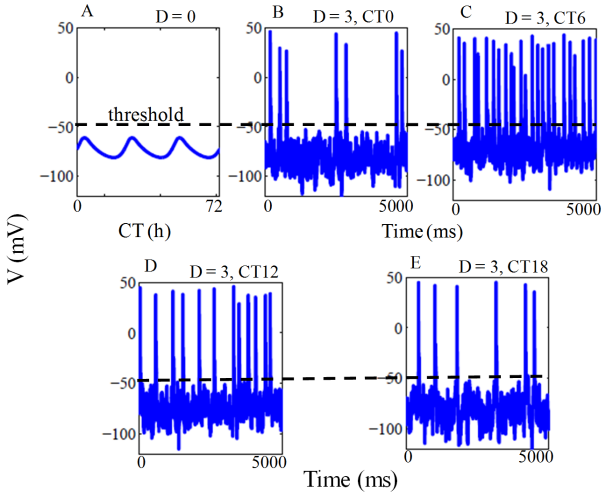


Figure 4: Simulations of the firing pattern in coupled GRN-HH-noise model. (A) Only subthreshold oscillations are observed in the absence of noise. (B-E) Noise-induced subthreshold oscillation that produces spikes at different circadian time. For the same noise intensity system shows different firing rates at various circadian phases. Black broken lines indicate the activation threshold of the HH model.

We examine the effect of noise at different circadian phases and the corresponding firing patterns for the noise intensity $D = 3$. Instead of performing stochastic simulation over continuous 24 h, we simulated the model only for desired circadian phases (CT0, 6, 12, and 18). At a particular circadian phase, we first set the integration time step to $0.02ms$ and run the system without adding the noise for one second. We then add the noise and perform the stochastic simulation for the desired time duration. Summary of the procedure for solving multiscale coupled oscillatory model is given in algorithm 1. The firing patterns recordings (5s in duration) at desired circadian times (CT_{des}) CT0, CT6, CT12, and CT18 are shown in Figure 4B-E. CT6 is the peaking time of GRN component M_P which is the mid of subjective day and CT18 is the subjective night. The number of spikes at CT18 is less compared to CT6, and this indicates that the firing frequency is less during subjective night compared to the subjective day as observed in the experiments. This important result indicates clearly that the slow varying circadian rhythm regulates the firing patterns.

Algorithm 1: Solve the multiscale coupled oscillatory model at desired circadian time to get the firing patterns

Input: multiscale coupled oscillatory model
Input: Gaussian noise w
Input: noise intensity D
Input: step size (Δt), Δt_1, Δt_2 ($\Delta t_1 \gg \Delta t_2$)
Input: window size ($window$) which compute from experiments
Input: desired circadian time, CT_{des}

If $CT \leq CT_{des}$
$\Delta t = \Delta t_1$
$D = 0$
else if $CT > CT_{des}$ & $CT < (CT_{des} + window)$
$\Delta t = \Delta t_2$
$D > 0$
end if
$v' = f(v) + D * w$ ——— (I)
print firing pattern according to (I)

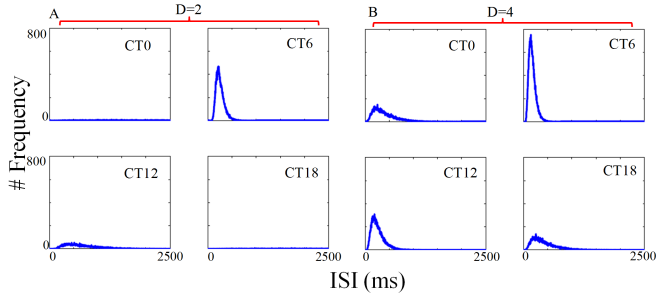


Figure 5: ISIH for GRN-HH-noise model. ISIs are randomly distributed at CT0, CT12, and CT18 for D = 2 (A) and D = 4 (B). At CT6 most of the firing periods is regular and concentrate near the intrinsic period.

ISIH for two other D values ($D = 2,4$) at different circadian phases are shown in Figure 5, and clearly, it unveils the difference in ISI distribution over the circadian time. Membrane voltage firing periods show a spread of distribution at CT0, CT12, and CT18, whereas a narrow distribution is exhibited at CT6, while the firing periods at CT6 are distributed near the intrinsic period of the system.

The value of conductance g_{N-leak} is also very near to Hopf bifurcation point at CT6 so that the HH system crosses the threshold easily and results in a narrow and high ISI distribution. However, when the g_{N-leak} value is far away from Hopf bifurcation point at CT0, CT12, and CT18 and hence the activation time is large resulting a wide and short ISI distribution.

3.3 Validation with experimental data

In order to verify the simulated firing data with experimentally observed circadian firing rate in SCN, we compute the firing frequency at different circadian time for different noise intensities. The results are shown Figure 6, which displays a clear circadian variation in the firing rates. During the subjective day, firing rates are between 3-10 Hz while during the subjective night it is found to be in the range 0-3 Hz under DD conditions. This is in excellent agreement with the experimental observation of Jones et al [3].

We also compute the power spectral density (PSD) of the membrane potential for various noise intensity at different circadian time, and the results are shown in Figure 7. Power spectral density of the membrane potential V was computed by averaging out 25 runs of the firing patterns of membrane potential with a duration of 5 min using fast Fourier transform. The height of the peak at CT6 increases with increase in noise intensity and the height starts to saturate and width of the peak increases for higher values of D (Figure 7B). Importantly, this trend is not observed at CT0, CT12, and CT18 (Figure 7A, C, D).

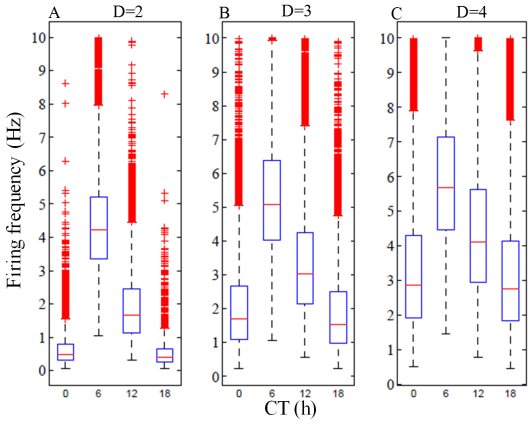


Figure 6: Circadian variation of firing frequency. Firing rates show circadian variation that is in good agreement with the experimental results [3]. CT6 is the peaking time of GRN component M_P which is the mid of subjective day and CT18 is the subjective night. On each box, the central mark indicates the median, and the top and bottom edges of the box indicates the 75^{th} and 25^{th} percentiles, respectively, and the outliers are plotted individually using the red '$+$' symbol.

We also quantify the system's response to noise at different circadian phases for which we calculate the SNR from power spectral density (Figure 8). At CT6, the model exhibits stochastic resonance for the optimum value of noise intensity $D = 2$ and at CT12 stochastic resonance occur for $D = 4$. We did not observe stochastic resonance at CT0 and CT18 for any value of D and presently we do not know why this the case. Taking together, the effect noise in SCN are circadian phase dependent and we speculate that stochastic resonance is circadian phase dependent because, at CT6, the peak time of the oscillator during the subjective day is close to the bifurcation point, whereas at other phases it is far away from the bifurcation point.

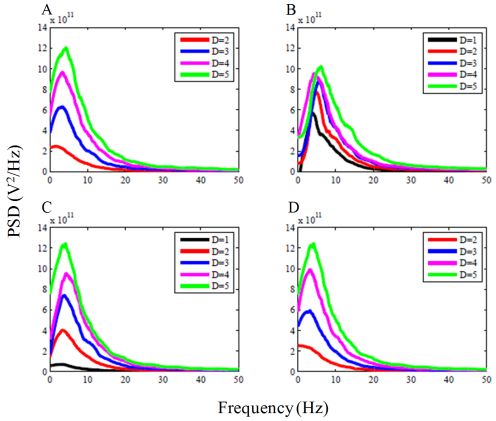


Figure 7: PSD for GRN-HH-noise model. With the increase of noise the height of the peak increases (A,C,D). However, at CT6 for a high value of noise, the growth of the height saturates and width of the peak increases (B).

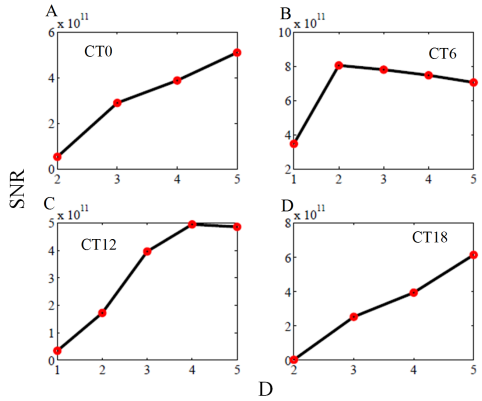


Figure 8: SNR for GRN-HH-noise model. At all circadian phases system did not exhibit stochastic resonance. (A, D) At CT0 and CT18, SNR increases with increase in the noise intensity, where $g_{Na-leak}$ value is far from the bifurcation point. (B, C) At CT6 and CT12 SNR is maximal for some value of noise intensity, exhibit stochastic resonance.

4 Conclusion

Previous studies have shown that the firing rate at SCN exhibit circadian rhythm with a period close to 24 h [3, 11]. However, the origin of firing pattern variations during subjective day and night are not known. Mechanisms of mutual regulation of slow varying gene regulatory networks and fast varying ionic channels in SCN are still under study and therefore, hardly there are any strong experimental evidence exists to understand the origin of firing pattern variations. Though channel noise plays an important role in inducing action potentials, this is inadequate to explain firing pattern variations that occur selectively at one frequency during day and night in a circadian phase dependent manner. We speculated and showed through multiscale simulations the role of noise and the slow varying circadian gene regulatory network in modulating the firing rates at different circadian phases by building coupled nonlinear ODE model that incorporates both GRN and electrophysiological part of SCN. Importantly, we find that the noise induced firing rate is circadian phase dependent, and the firing rate is higher/lower during subjective day/night respectively. This is also in excellent agreement with the experimental observations. From the dynamical system point of view, we hypothesize that the firing is rapid during subjective day because the peaking of *per* gene occur during the subjective day which in turn takes the ionic channel close to the bifurcation point and thereby facilitates noise induces coherent firing pattern. On the other hand, when *per* concentration during subjective night is at nadir/trough the ionic channel is far away from the bifurcation point and therefore, firing rate is subdued and noisy. We have also not looked into the role of window-size and the window-size we took is based on the experimental data and in this window-size, only during certain circadian phase, stochastic resonance is seen. Presently, we do not have any explanation why stochastic resonance is seen only in select circadian phases and not in other phases. We intend to look into this aspect closely in the future work. We are particularly interested in modeling the firing pattern variations from the morning and evening oscillator point of view for which the mathematical model of morning and evening oscillators are already studied detail [12]. In future, we would also like to extend this work further to study thoroughly the role of stochastic resonance in circadian rhythms.

Acknowledgment

This work is supported by the DST cognitive neuroscience grant, SR/CSI/299/2012 (awarded to KS) from the Department of Science and Technology, INDIA.

References

[1] J. S. Takahashi, "Transcriptional architecture of the mammalian circadian clock," *Nature Reviews Genetics*, vol. 18, pp. 164–179, 2016.

[2] Z. G. Jiang, Y. Yang, Z. P. Liu, and C. N. Allen, "Membrane properties and synaptic inputs of suprachiasmatic nucleus neurons in rat brain slices," *The Journal of Physiology*, vol. 499, no. 1, pp. 141–159, 1997.

[3] J. R. Jones and D. G. McMahon, "The core clock gene per1 phases molecular and electrical circadian rhythms in SCN neurons," *PeerJ*, vol. 4, p. e2297, 2016.

[4] R. Benzi, A. Sutera, and A. Vulpiani, "The mechanism of stochastic resonance," *Journal of Physics A: mathematical and general*, vol. 14, no. 11, p. L453, 1981.

[5] H. Gang, T. Ditzinger, C. Ning, and H. Haken, "Stochastic resonance without external periodic force," *Physical Review Letters*, vol. 71, no. 6, p. 807, 1993.

[6] A. S. Pikovsky and J. Kurths, "Coherence resonance in a noise-driven excitable system," *Physical Review Letters*, vol. 78, no. 5, p. 775, 1997.

[7] C. O. Diekman, M. D. Belle, R. P. Irwin, C. N. Allen, H. D. Piggins, and D. B. Forger, "Causes and consequences of hyperexcitation in central clock neurons," *PLoS Computational Biology*, vol. 9, no. 8, p. e1003196, 2013.

[8] B. C. Goodwin, "Oscillatory behavior in enzymatic control processes," *Advances in Enzyme Regulation*, vol. 3, no. Supplement C, pp. 425 – 437, 1965.

[9] B. Ermentrout, *Simulating, analyzing, and animating dynamical systems: a guide to XPPAUT for researchers and students*. SIAM, 2002.

[10] P. Kloeden and E. Platen, *Numerical Solution of Stochastic Differential Equations*. Stochastic Modelling and Applied Probability, Springer Berlin Heidelberg, 2011.

[11] C. S. Colwell, "Linking neural activity and molecular oscillations in the SCN," *Nature Reviews Neuroscience*, vol. 12, no. 10, pp. 553–569, 2011.

[12] S. Shiju and K. Sriram, "Hypothesis driven single cell dual oscillator mathematical model of circadian rhythms," *PloS one*, vol. 12, no. 5, p. e0177197, 2017.

cMutant : A Web Server and Compute Pipeline for Exploring the Effects of Amino Acid Substitutions via Rigidity Mutation Maps

Hunter Read, Kyle Daling, Connor Freitas, Filip Jagodzinski*
Western Washington University
Bellingham, WA, 98225, USA
filip.jagodzinski@wwu.edu

Abstract

Pharmaceutical companies rely on the ability to analyze the effects of protein mutations to develop medicines for treating a variety of diseases. Although mutagenesis experiments performed in a physical protein can provide insights about the role of a single amino acid, such experiments are laboriously difficult and may require months of wet lab work. Consequently conducting exhaustive mutagenesis screens which involve mutating all residues to all other amino acids, is impractical. To help guide such wet lab experiments, computational approaches are available, but most do not permit an exhaustive screening of all residues and their impact on a protein when mutated. For this work we have integrated into a compute pipeline and server our *in silico* mutation analysis method for quickly generating protein variants. We leverage a quick computational algorithm to assess the rigidity of the wild type and mutants, and use the results to infer which residues are most sensitive to an amino acid substitution. Our server and pipeline leverage concurrency principles permitting an exhaustive screening of all mutations for all residues in a protein in as little as a few minutes. We report here on the performance and utility of the pipeline, and present a case study to highlight the utility of Mutation Maps generated by our server, cMutant, available at `https://cmutant.cs.wwu.edu/`.

Introduction

Experimentalists mutate and analyze proteins to develop better medicine for treating a wide range of diseases [27]. Conducting mutation analyses in a physical protein can require months of wet-lab work, with the aim to provide information to help engineer pharmaceutical drugs targeting specific proteins [22].

A variety of computational approaches and *in silico* protein mutation analysis tools aim to provide a screen to help guide wet lab experimentalists where they might focus their attention for conducting mutagenesis experiments on physical proteins. The majority of most existing screening software tools permit exploring the effect of only a single mutation at one specific residue in a protein, while the few approaches that permit exhaustive *in silico* studies for a protein have a variety of limitations due to their dependencies on homology or energetics data that may not always be available.

In our previous work [7, 2], we have motivated the use of a fast combinatorial approach called rigidity analysis, in combination with our custom *in silico* mutation engine for generating mutant structure files, in assessing the effects of amino acid substitutions.

For this work, we present a compute pipeline and publicly available server, cMutant, that relies on concurrency principles to greatly reduce the runtime of performing an exhaustive mutation screen for all residues in a protein. We reduce the runtime of *in silico* mutation experiments from days to hours – and sometimes to minutes. We achieve such a speedup by executing our pipeline concurrently on multiple cores available on our server. To permit a user to perform a visual inspection of the effects of the exhaustive mutation experiments, we generate a mutation map which is presented via a graphical user interface, and which is stored in a database for future retrieval. The utility of our mutation maps we have demonstrated in our previous work [28].

Related Work

To help complement and inform wet lab work, various modeling and computational methods, including some available via web servers, are available. They strive to predict the effects of mutations. Early algorithms ranged from those that searched for best side-chain conformations as a measure of the impact of a mutation [6, 16, 25], to those that relied on heuristic energy functions [10, 19]. Yet others relied on large data sets of homologous proteins [30, 3, 31]. More recently, machine learning (ML) approaches have gained notoriety, with some having high prediction rates upwards of 80% [4, 14, 17, 20]. However, the energy-, homology- and ML-based approached have several limitations. Many of

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

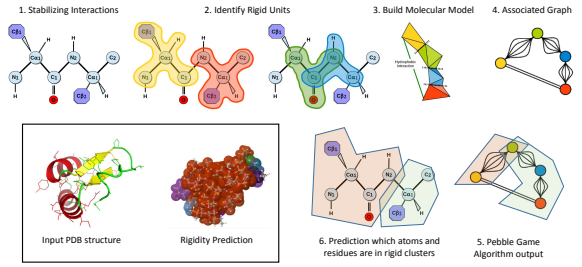


Figure 1: Rigidity analysis involves modeling a biomolecule as a mechanical model, which is analyzed using an efficient pebble game algorithms. The results are used to infer the rigid and flexible regions of a biomolecule.

them are dependent on large data sets [21, 32], some require costly energy calculations [5, 24, 26], and others still are dependent on free energy calculations as well as access to propensity tables [23], data which is not always available, or which is computationally costly to calculate.

In our previous work, we developed several computational approaches for quickly generating large data sets of *in silico* mutants. Incipient experiments enabled mutating a residue to only one of Alanine, Glycine, or Serine [15], but more recently our mutation software has been expanded to permit *in silico* mutating a residue to all possible other amino acids [2].

To help reason about the effects of mutations, we take an approach that does not rely on propensity tables, costly energy calculations, nor is dependent on homology data. Instead we rely on a fast combinatorial approach for assessing the rigidity of a protein [9, 13]. In rigidity analysis, atoms and their chemical interactions are used to construct a mechanical model. A graph is constructed from the model, and pebble game algorithms [12] are used to analyze the rigidity of the associated graph. The results are used to infer the rigid and flexible regions of the protein (Figure 1).

Rigidity Distance

In this work cMutant compares the rigidity analysis results of the wild type (WT), non-mutated form of a protein, to the rigidity analysis results of the mutant. This builds on our previous work [1, 8], in which we developed and utilized a$RD_{WT \rightarrow mutant}$ rigidity distance metric to quantitatively assess the impact of mutating a residue to one of the other 19 naturally occurring amino acids:

$$RD_{WT \rightarrow mutant} : \sum_{i=1}^{i=LRC} i \times [WT_i - Mut_i]$$

where WT refers to Wild Type, Mut refers to mutant,

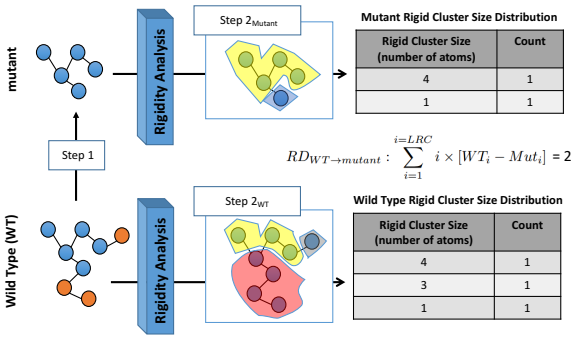


Figure 2: Comparing the rigid cluster distributions (sizes and counts) for the Wild Type and Mutant structures enables assessing quantitatively the effect of an amino acid substitution via the Rigidity Distance $RD_{WT \rightarrow mutant}$ metric.

LRC is the size of the Largest Rigid Cluster (in atoms). Each successive summation term of the $RD_{WT \rightarrow mutant}$ metric calculates the difference in the count of a specific cluster size, i, of the wild type and mutant, and weighs that difference by i (Sample in Figure 2).

Server & Software Design

Our contributions for this work includes a concurrent implementation of our mutation and analysis software and the auto-generation of mutation maps [28] to aid in the visual analysis of an exhaustive mutation screen. In this section we describe the server and compute pipeline, as well as the analysis methodology that culminates in a mutation map.

Overview

cMutant offers features that are not available via other tools and web services. Upon invocation, the server generates all mutant structures as asked-for by the user via the front-end. The infrastructure leverages principles from concurrency theory to vastly reduce the execution time needed for conducting exhaustive mutation experiments. cMutant offers a graphical user interface (GUI), that enables a user to view all mutations via a mutation map which permits a user to investigate individual point mutations and download specific results. The system design is summarized in Figure 3.

Back-End Infrastructure

The computational infrastructure integrates a variety of our in-house custom software, as well as off-the shelf

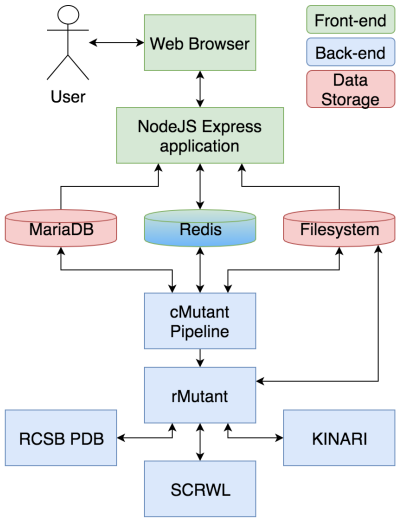


Figure 3: cMutant includes front-end (GUI) and back-end functionality, enabling a user to interface with our custom mutation and analysis software.

and freely available tools. These include KINARI [9] and ProMuteHT[2], along with SCWRL [18]. The pipeline is invoked when a user interacts with the GUI to specify the PDB ID (protein structure file), along with parameters designating which residues are to be mutated. Use of concurrency principles enabled by the threading capabilities of the multi-core server allows for each unique *in silico* point mutation to be invoked in a separate thread. The count of threads is limited by the number of available processors, and the output data files of each experiment are stored to files locally on the server for archiving and retrieval by the user.

The back-end infrastructure performs concurrent execution of KINARI and proMuteHT for quickly generating and processing of a large set of protein mutants (Table 1). cMutant is able to decrease exhaustive protein mutation run times by a factor equal to that of the number of cores available on the server, with additional speed-up obtained through allowing mutations to take advantage of the pipelining ability of the CPU architecture. Each experiment requires analyzing the wild type protein once, before any *in silico* protein mutations are performed. This first step is not run concurrently. Run-times were determined by clock time at initialization of the compute pipeline through execution of all intermediate steps, until pipeline termination resulting in a mutation map.

Front-End Infrastructure

The front-end GUI of cMutant includes an **Experiment (and Results)** section. There users specify experiment parameters, and view results as they become

Table 1: Run-times (minutes) for threaded (thread) and serial (ser) invocations of cMutant, and speedup ratios (sr) resulting from use of concurrency. # res=num. of residues; # muts=num. mutants generated.

PDB File	# res	# muts	thread	ser	sr
1PLW	5	100	0.65	4.37	6.72
1DPK	20	400	3.32	22.7	6.84
2LK0	30	600	7.35	45.3	6.16
1HN3	40	800	10.8	65.4	6.06
1YUG	50	1000	19.2	103	5.39
5NHQ	71	1420	36.9	190	5.15
1A1Z	83	1660	63.7	301	4.72
1HHP	99	1980	95.9	426	4.44

available. A **Retrieve Experiments** section permits retrieving data from past computation runs, as well as viewing the current server load.

The **Experiment** section offers a GUI (Figure 4) with options for a user to:

(1) specify a PDB ID for which a mutation screen is to be performed

(2) which residues to mutate (an **all** option is available for designating an exhaustive screen)

(3) specifying what each selected residue(s) should be mutated to, for which an **all** options is also available.

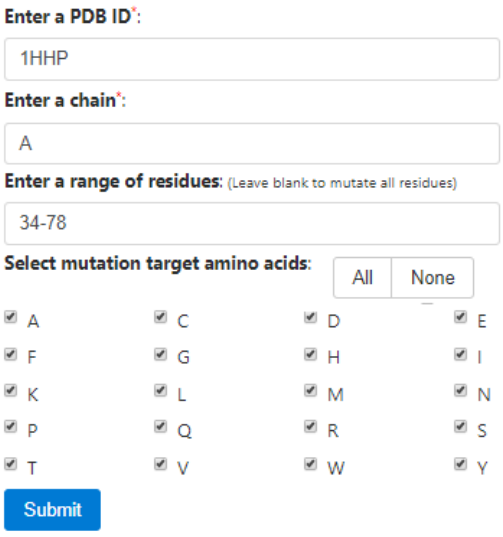


Figure 4: cMutant's GUI offers the option to specify an exhaustive mutation screen, or to mutate a subset of the residues (range of residues). Each selected residue can be mutated to all other possible amino acids, or a custom subset (mutation targets).

When an experiment is initiated, a user is provided with an alphanumeric experiment ID which can be used for later retrieval of the experiment data which is stored to a database.

Server-side technologies such as NodeJS and ExpressJS provide the functionality for transmitting data between the back-end software and GUI. As data is being generated by the multiple threads that are invoked, the results are displayed and updated in real time. Communication between the cMutant pipeline and GUI is accomplished by using the in-memory data structure store Redis.

The **Result** pane presents a Mutation Map, which is a heat map generated from the distance metric values (See section Rigidity Distance) computed for each residue that was mutated. A full explanation can be found in [28]. The color in each cell in a Mutation Map corresponds to a rigidity distance, which is a measure, based on the rigid clusters of the mutant and wild type. A user can mouse-over a specific cell in the Mutation Map to view the rigidity distance score for that residue, or to download the data for that specific *in silico* mutation. A rigidity distance far greater or far less than zero indicates that the mutant is structurally vastly different than the wild type, while a rigidity distance score near zero specifies that the wild type and mutant are structurally similar, as inferred using the rigidity cluster data. The magnitude of the rigidity distance can be used to indirectly infer the magnitude of the impact of an amino acid substitution.

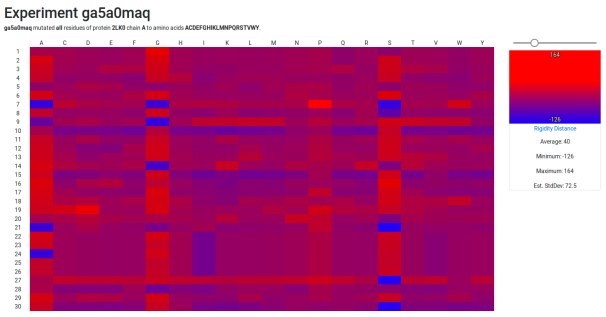


Figure 5: Mutation Maps : for each residue number (y-axis), a color at each target residue (x-axis) specifies the rigidity distance metric score for a mutation.

A sample Experiment results pane, for experiment `ga5a0maq`, is shown in Figure 5. That mutation map is for an exhaustive *in silico* mutation screen for the 30-residues PDB file 2LK0, which is the structure of a RanBP2-type zinc finger of RBM5. A dynamically updated color legend indicates that a red cell has a high rigidity distance, while a blue cell has a low rigidity distance score, and that the average, minimum, and maximum Rigidity Distance scores are 40, -126, and 164. Most telling in the Mutation Map for 2LK0 is that specific residues upon their *in silico* mutation to certain residues yield very low (highly negative), or very high (highly positive) rigidity distance scores. A very low rigidity distance score for a residue's mutation to a specific amino acid indicates that that mutation results in a mutant that has far more large rigid clusters than the WT. Such a mutation can be inferred to be stabilizing. The converse is true for very high positive rigidity scores. In the case of 2LK0, using the Mutation Map, the blue spots identify that residues 7, 9, 14, 21, 24, 27, and 30, have strong stabilizing effects on the protein as inferred using rigidity analysis.

Case Study, 1HHP

To assess the speed and usefulness of cMutant, we exhaustively *in silico* mutated all residues of PDB structure 1HHP, which is the monomeric form of the 99 amino acid HIV-1 Protease. A zoomed in portion (residues 15 to 40) of the Mutation Map for 1HHP is shown in Figure 6.

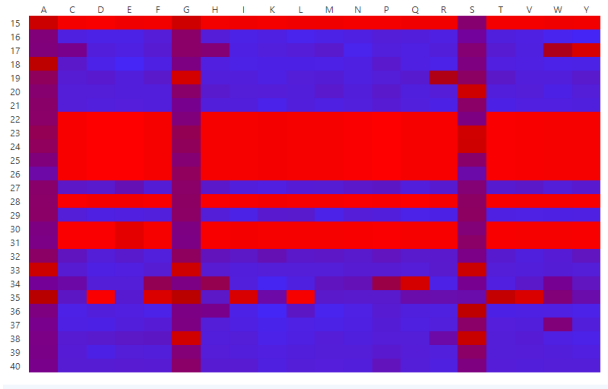


Figure 6: Zoomed in Mutation Map for 1HHP, residues 15-40. Residues 22-26, as well as 28, and 30 and 31 are especially sensitive to mutations as evidenced by the red Rigidity Distance scores for nearly all mutations performed at those residues.

Residues 24-26 of HIV-1 Protease constitute a catalytic triad, the active site of the protein, on which a host of wet lab experiments have been conducted and for which there is a lengthy literature [11, 29]. The residues near the active site of HIV-1 Protease are known to be critical to the protein's function, and indeed are highly resistant to mutations. Specific residues at those locations must be present in order for the protein to perform its catalytic function. As a first proof-of-concept result, we consider it encouraging that cMutant identified those residues near the active site as being least resistant to mutations, because *in silico*

mutations performed on them in nearly all cases highly disrupted the protein's structure. See [28] for a more detailed example of the utility and use, including a box plot analysis, of Mutation Maps.

Future and On-Going Work

Future and-going work on cMutant involves three main avenues, including 1) improving the server's speed by leveraging additional concurrency principles, 2) adding additional front-end GUI features, and 3) assessing and improving the accuracy of the predictions doled up by the Mutation Map. In our most recent work, we have developed machine learning models capable of predicting at up to 80% accuracy the effect of mutations [8]. That predictive capability is being integrated into cMutant.

For improving the GUI, we are developing additional UI elements to allow the user to quickly access important trends and details of the results from a computation experiment run. In addition to the mutant and WT structure files, along with the rigidity data, for each cell in a Mutation map that can be currently downloaded, we aim to integrate a protein viewer visualization engine that will color code the 3-D surface of a protein to display rigidity metrics of those residues on the surface.

A current limitation of the server is that it is able to perform exhaustive mutation screens for single chain proteins only. Current work in our lab has culminated in an improved mutation engine, ProMuteHT, which is being integrated into the cMutant pipeline allowing it to reason about any protein in the PDB.

For further validation of the use of Mutation Maps beyond what we have reported previously [28], we are correlating our rigidity distance scores for point mutations against $\Delta\Delta G$ data attained from experiments on physical proteins, which gives empirical evidence of the effects of mutations. We are tallying Pearson Correlation coefficients, and aim to supplement the Mutation Map data with that information.

Conclusions

We have developed a compute pipeline and server, cMutant, for performing a rigidity-based mutation screen that exhaustively generates and analyzes all possible mutant structures with a single amino acid substitution. We achieve fast run-times by leveraging concurrency principles, and also generate a Mutation Map which aids in a visual analysis enabling identification of residues that are highly sensitive to mutations. We present a case study for HIV-1 Protease, and correlate our interpretation of the analysis of the Mutation Map with known biological properties of the protein's active site.

Acknowledgments

HR designed and deployed the database, and developed the pipeline enabling the rigidity and proMuteHT software to run concurrently. KD developed the web server code used for serving the GUI and transmitting data to it. CF designed the results page and mutation map functions. FJ supervised the work. All authors contributed to writing the manuscript.

References

[1] E. Andersson, R. Hsieh, H. Szeto, R. Farhoodi N. Haspel, and F. Jagodzinski. Assessing how multiple mutations affect protein stability using rigid cluster size distributions. In *Computational Advances in Bio and Medical Sciences (ICCABS), 2016 IEEE 6th International Conference on*, pages 1–6. IEEE, 2016.

[2] Erik Andersson and Filip Jagodzinski. ProMuteHT: A high throughput compute pipeline for generating protein mutants in silico. In *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 655–660. ACM, 2017.

[3] Jeffrey R Brender and Yang Zhang. Predicting the effect of mutations on protein-protein binding interactions through structure-based interface profiles. *PLoS Comput Biol*, 11(10):e1004494, 2015.

[4] J. Cheng, A. Randall, and P. Baldi. Prediction of protein stability changes for single-site mutations using support vector machines. *PROTEINS: Structure, Function, and Bioinformatics*, 62:1125–1132, 2006.

[5] Y Dehouck, JM Kwasigroch, M Gilis, and Rooman M. Popmusic 2.1: a web server for the estimation of protein stability changes upon mutation and sequence optimality. *BMC Bioinformatics*, 12, 2011.

[6] R.L. Jr. Dunbrack and M. Karplus. Conformational analysis of the backbone-dependent rotamer preferences of protein sidechains. *Nature Structural Biology*, 1:334–340, 1994.

[7] I. Streinu F. Jagodzinski, J. Hardy. Using rigidity analysis to probe mutation-induced structural changes in proteins. *Journal of Bioinformatics, Computational Biology*, 10:1242010–1242027, 2012.

[8] Roshanak Farhoodi, Max Shelbourne, Rebecca Hsieh, Nurit Haspel, Brian Hutchinson, and Filip Jagodzinski. Predicting the effect of point mutations on protein structural stability. In *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 247–252. ACM, 2017.

[9] N. Fox, F. Jagodzinski, and I. Streinu. KINARI-Lib: a C++ library for pebble game rigidity analysis of mechanical models. In *Minisymposium on Publicly Available Geometric/Topological Software, Chapel Hill, NC, USA*, June 2012.

[10] D. Gilis and M. Rooman. Predicting protein stability changes upon mutation using database-dervied potentials: Solvent accessibility determines the importance of local versus non-local interactions along the sequence. *Journal of Molecular Biology*, 272(2):276–290, 1997.

[11] Viktor Hornak, Asim Okur, Robert C Rizzo, and Carlos Simmerling. Hiv-1 protease flaps spontaneously open and reclose in molecular dynamics simulations. *Proceedings of the National Academy of Sciences of the United States of America*, 103(4):915–920, 2006.

[12] D.J. Jacobs and B. Hendrickson. An algorithm for two-dimensional rigidity percolation: the pebble game. *Journal of Computational Physics*, 137:346–365, 1997.

[13] D.J. Jacobs, A.J. Rader, M.F. Thorpe, and L.A. Kuhn. Protein flexibility predictions using graph theory. *Proteins 44*, pages 150–165, 2001.

[14] F. Jagodzinski, B. Akbal-Delibas, and N. Haspel. An evolutionary conservation & rigidity analysis machine learning approach for detecting critical protein residues. In *CSBW (Computational Structural Bioinformatics Workshop), in proc. of ACM-BCB (ACM International conference on Bioinformatics and Computational Biology)*, pages 780–786, September 2013.

[15] F. Jagodzinski, J. Hardy, and I. Streinu. Using rigidity analysis to probe mutation-induced structural changes in proteins. *Journal of Bioinformatics and Computational Biology*, 10(3), 2012.

[16] J Janin and S Wodak. Conformation of amino acid side-chains in proteins. *J Mol Biol*, 125(3):357–386, Nov 1978.

[17] Lei Jia, Ramya Yarlagadda, and Charles C Reed. Structure based thermostability prediction models for protein single point mutations with machine learning tools. *PloS one*, 10(9):e0138022, 2015.

[18] Georgii G Krivov, Maxim V Shapovalov, and Roland L Dunbrack. Improved prediction of protein side-chain conformations with scwrl4. *Proteins: Structure, Function, and Bioinformatics*, 77(4):778–795, 2009.

[19] C. Lee and M. Levitt. Accurate prediction of the stability and activity effects of site-directed mutagenesis on a protein core. *Nature*, 352:448–451, 1991.

[20] Yunqi Li and Jianwen Fang. Prots-rf: a robust model for predicting mutation-induced protein stability changes. *PloS one*, 7(10):e47247, 2012.

[21] Majid Masso and Iosif I Vaisman. Auto-mute: web-based tools for predicting stability changes in proteins due to single amino acid replacements. *Protein Engineering Design and Selection*, 23(8):683–687, 2010.

[22] Tatiana Maximova, Ryan Moffatt, Buyong Ma, Ruth Nussinov, and Amarda Shehu. Principles and overview of sampling methods for modeling macromolecular structure and dynamics. *PLoS computational biology*, 12(4):e1004619, 2016.

[23] Caitlyn L McCafferty and Yuri V Sergeev. In silico mapping of protein unfolding mutations for inherited disease. *Scientific Reports*, 6:37298, 2016.

[24] Vijaya Parthiban, M. Michael Gromiha, and Dietmar Schomburg. Cupsat: prediction of protein stability upon point mutations. *Nucleic Acids Research*, 34(suppl 2):W239–W242, 2006.

[25] J.W. Ponder and F.M. Richards. Tertiary templates for proteins: Use of packing criteria in the enumeration of allowed sequences for different structural classes. *Journal Molecular Biology*, 193:775–791, 1987.

[26] Lijun Quan, Qiang Lv, and Yang Zhang. Strum: structure-based prediction of protein stability changes upon single-point mutation. *Bioinformatics*, 32(19):2936–2946, 2016.

[27] Boris Reva, Yevgeniy Antipin, and Chris Sander. Predicting the functional impact of protein mutations: application to cancer genomics. *Nucleic acids research*, 39(17):e118–e118, 2011.

[28] Michael Siderius and Filip Jagodzinski. Mutation sensitivity maps: Identifying residue substitutions that impact protein structure via a rigidity analysis in silico mutation approach. *Journal of Computational Biology*, 25(1):89–102, 2018.

[29] Sergio Filipe Sousa, Bruno Tamames, Pedro Alexandrino Fernandes, and Maria Joao Ramos. Detailed atomistic analysis of the hiv-1 protease interface. *The Journal of Physical Chemistry B*, 115(21):7045–7057, 2011.

[30] C.M. Topham, N. Srinivasan, and T. Blundell. Prediction of the stability of protein mutants based on structural environment-dependent amino acid substitutions and propensity tables. *Protein Engineering*, 10(1):7–21, 1997.

[31] C.L. Worth, R. Preissner, and L. Blundell. Sdm-a server for predicting effects of mutations on protein stability and malfunction. *Nucleic Acids Research*, 39(Web Server Issue):W215–W222, 2011.

[32] Hongyi Zhou and Yaoqi Zhou. Distance-scaled, finite ideal-gas reference state improves structure-derived potentials of mean force for structure selection and stability prediction. *Protein science*, 11(11):2714–2726, 2002.

Integration of Biomedical Big Data Requires Efficient Batch Effect Reduction

Jane Synnergren
Systems Biology Research Center
University of Skövde
SE-541 28, Skövde, Sweden
jane.synnergren@his.se

Nidal Ghosheh
Systems Biology Research Centre
University of Skövde
SE-541 28, Skövde, Sweden
nidal.ghosheh@his.se

Pierre Dönnes
SciCross AB
Gothia Science Park
SE-541 23, Skövde, Sweden
pierre@scicross.com

Abstract

Efficiency in dealing with batch effects will be the next frontier in large-scale biological data analysis, particularly when involving the integration of different types of datasets. Large-scale omics techniques have quickly developed during the last decade and huge amounts of data are now generated, which has started to revolutionize the area of medical research. With the increase in the volume of data across the whole spectrum of biology, problems related to data analytics are continuously increasing as analysis and interpretation of these large volumes of molecular data has become a real challenge. Tremendous efforts have been made to obtain data from various molecular levels and the most recent trends show that more and more researchers now are trying to integrate data of various molecular types to inform hypotheses and biological questions. Tightly connected to this work are the batch-related biases that commonly are apparent between different datasets, but these problems are often not tackled. In present study the ComBat algorithm was applied and evaluated on two different data integration problems. Results show that the batch effects present in the integrated datasets efficiently could be removed by applying the ComBat algorithm.

1. Introduction

Effective integration and analysis of high-throughput data are expected to deliver novel clinical insights and promising therapeutic options. However, technical variation inherent in these datasets makes the integration task problematic and dealing with technical heterogeneity or batch effects during data integration has proven to be a real challenge [1]. Nevertheless, to combine datasets without adjusting for batch effects is in general inappropriate with risks of misinterpretation of results. Given the large amount of data that are generated daily and that need to be considered synergistically instead of independently, batch-effect mitigation is foreseen to become the next big challenge in mega biological data analysis. Without effective management, it is impossible to take full advantage of the vast amount of information already available, and use it synergistically for building better classifiers, performing better functional analysis, and producing clinically useful outcomes [1].

1.1. Sources of batch effects

Systematic non-biological differences between different datasets are commonly referred to as technical variation, also termed "batch effects" [2]. There are many different sources of variations in biomedical data, regardless if dealing with large-scale or small-scale datasets. Many different platforms are available for data generation and different techniques are used. However, the lack of standardization makes it challenging to compare and integrate data from the different technologies due to the introduced platform-dependent systematic variation. Several studies have demonstrated that pooling data derived from different platforms is a complex task with numerous pitfalls. In addition to platform variation there are also other factors known to introduce systematic variation for example analysis conducted at different sites, by alternating personnel, differences in experiments, reagents, instruments, and lots, or simple day-to-day variation, and all these are well-known factors that inevitably result in introduction of batch differences [1, 2].

1.2. Microarray data

The microarray technology has been extensively used for genome-wide gene expression analysis for almost two decades now and is a well-established, cost-effective, high-throughput technology that makes it possible to measure the expression levels of thousands of genes simultaneously, and thus offers an efficient way to generate a snapshot of the entire transcriptome [3]. The workflow of a microarray transcription analyses in itself consists of multiple steps such as RNA extraction, labeling, hybridization, washing, and scanning, and each of these steps is a potential risk of introducing variation in the data. I addition, several different microarray platforms are commercially available, and these differ with respect to for example the fabrication methodologies and length of the oligonucleotide probes. There are huge amounts of microarray data available in public biological databases such as ArrayExpress [4] and GEO database [5]. To fully utilize these enormous resources effective and reliable

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

methods for both normalization and reduction of batch effects are required, which manage to reduce the non-biological variation while maintaining the true biological differences in the data. Although newer techniques such as next generation sequencing in many cases now are replacing the microarray technology, microarrays are still routinely used in many different types of large-scale analysis due to its well-established nature, cost-effectiveness, and the availability of extensively developed analysis frameworks [6].

1.3. The need for integration of datasets

The rapid development of advanced techniques for generation of biomedical big data has turned life science research into a data driven field. The generation of data is no longer a challenge and terabytes of data can be produced at relatively low investment costs. The data available in public biological databases for peer researchers to use in their research increases rapidly and constitute a valuable resource, which also facilitates larger meta-studies in which many datasets are integrated and used in combination. The term *data integration* refers to the situation where, for a given system, multiple sources (and possible types) of data are available and we want to study them integratively to improve knowledge discovery [7]. Integration of gene expression data obtained from multiple experiments provides opportunities to increase the statistical power of the analyses since commonly, these datasets in isolation are characterized by a low number of samples and a high number of variables. Thus, integrating several smaller datasets theoretically boosts power and better reflects the underlying population. However, proper integration requires resolving the technical heterogeneity, including batch effects.

1.4. Batch effects reduction methods

Several batch effect correction algorithms are available but there is still limited knowledge about effective batch effect mitigation, and new batch effect-associated problems are still emerging [1]. These include false effects due to misapplying batch effect correction algorithms and positive bias during model evaluations. Depending on the choice of algorithm and experimental setup, biological heterogeneity can be mistaken for batch effects and wrongly removed [1]. The batch reduction methods can be classified based on their area of use. In the simple linear models a biological feature is modeled as a linear combination of class and batch effects. Examples of simple linear model methods are mean-scaling and zero-centering. Another class of methods are the ones that use Bayesian inference to estimate biological features in a batch, e.g. the ComBat method which applies the empirical Bayes approach [8]. These two classes of methods described above require that all known batch factors are specified to make a reliable batch effect estimation. In contrast to these methods other approaches use the full data matrix for estimation of batch-related against class-related variation. Examples of methods in this class are the Surrogate Variable Analysis (SVA) and the Removed Unwanted Variation (RUV) methods. The SVA method first requires the specification of the class factor and assumes that consistent sources of variation not associated with the class factor are likely associated with some unknown batch factor. The method then estimates the batch effect via singular value decomposition and removes the batch effect from data using regression [9]. The RUV method is similar to SVA, but incorporates information from endogenous control genes or i.e. housekeeping genes, which are expected to be unaffected by class effects and therefore used to estimate batch effects [10]. However, specifying housekeeping genes can be controversial and, in some cases, so-called 'housekeeping genes' are directly related to disease or tissue specific expression and therefore inappropriate to use for batch effect estimations [1].

1.5. The ComBat algorithm

In the present study the ComBat algorithm has been used for batch effects reduction and this method is based on the empirical Bayes (EB) method. EB methods are robust for adjustment of batch effects in data whose batch sizes are small and is very appealing in various microarray problems because of their ability to robustly handle high-dimensional data with small sample sizes. These methods are typically designed to "borrow information" across genes and experimental conditions in hope that the borrowed information will lead to better estimates or more stable inferences [8]. The EB methods have usually been designed to stabilize the expression ratios for genes with very high or very low ratios, stabilize gene variances by shrinking variances across all other genes, possibly protecting their inference from artifacts in the data [8].

Systematic batch biases common across genes are incorporated in making adjustments, assuming that phenomena resulting in batch effects often affect many genes in similar ways, for example with increased expression or higher variability. The ComBat method estimates the model parameters that represent the batch effects, by "pooling information" across genes in each batch to "shrink" the batch effect parameter estimates toward the overall mean of the batch effect estimates across all genes. These EB estimates are then used to adjust the data for batch effects, providing more robust adjustments for the batch effect on each gene [8]. The method is divided into three steps, (i) standardization of the data, (ii) estimation of EB batch parameters using empirical priors, and (iii) batch effect adjustment, as described in detail in [8].

1.6. Perspectives of omics data integration

As compared to studies of a single omics type, multi-omics data offers the opportunity to better understand the flow of information that for example underlies various diseases. Integrating omics datasets are thus expected to provide additional guidance in e.g. personalization of treatments [11]. Moreover, multi-omics data provides promising opportunities to increase the understanding of the regulatory mechanisms in the cells and the functions of genes and proteins. However, standardized methods for performing multi-omics data integration is still in its infancy. Due to the high level of complexity of these datasets, multi-omics data integration is considered one of the major future challenges in this era of precision medicine [11]. Integrative omics approaches often rely on a "holistic" view, which attempts to interrogate a sufficiently large number of individuals and incorporate the many sources of variability into statistical models [12]. Thus, a crucial aspect for success of integrative omics studies is the availability of large datasets of various molecular types, and high-throughput techniques for large-scale data generation have made collection of such data feasible. In this context appropriate batch effect reduction methods that also can handle different types of data (genomics, transcriptomics, proteomics, and/or metabolomics) will be critical for proper multi-omics data integration.

2. Methods

In this paper we applied the ComBat algorithm and investigated the performance of this method for removal of batch effects in two different study setups. In these setups the combination of multiple datasets was required in order to perform the subsequent expression analysis. In the first dataset, the batches of data were generated at different time points but with the same molecular technique. In the second study design, different molecular techniques were used to generate the two batches of data and they were also generated at different time points.

2.1 The cardiac biopsy study

This study extends our previous work on cardiac biopsies where transcriptional profiling of biopsies from two different cardiac locations of five different male patients that were under cardiac surgery was performed [13]. The purpose of that study was to investigate differences in gene expression between left ventricular (LV) and right atrial (RA) cardiac tissue. Interesting transcriptional differences between these two locations were reported and next step was to extend this study to also include female patients to be able not only to analyze location dependent transcriptional differences but to explore putative differences related to the gender of the patient as well. Thus, the same experimental setting was repeated on five additional female patients resulting in two datasets from paired biopsies from the LV and the RA. Transcriptional analysis was performed on the biopsy samples using the whole transcript Gene ST 1.0 arrays (Affymetrix, www.affymetrix.com). Since the same technical platform was used for both these experiments, minimal variability attributed to technical aspects was expected. However, the experiments were performed at different time point and slightly different procedures were applied for the cDNA synthesis. These differences, or the fact that the experiments were run at different time points, or a combination of both, introduced large batch effects that complicated the downstream bioinformatics analysis of the data. In this study design the known putative batch effects were *differences in time point* for performance of the wet-lab experiments and slightly *different procedure for the cDNA synthesis*. Quality control of the datasets revealed large batch effects that needed to be adjusted for.

2.2 The hepatic differentiation study

In the second study design data from hepatic differentiation of human pluripotent stem cells (hPSCs) was analyzed with respect to transcriptional patterns during human hepatic development. The main study included cells harvested at defined time points during the differentiation of the hPSCs towards the hepatic lineage [14, 15]. Four different time points were sampled including day 0, day 5, day 14, day 25, and two reference samples from adult liver tissue (AL) where included as controls. The differentiation experiments were repeated for six different stem cell lines, and for each cell line, duplicated samples were analyzed. No fetal liver (FL) samples were available for the project by the time of the transcriptional analysis. However, FL would have been a relevant control since the stem cell derived hepatocytes are known to have an immature phenotype. To still be able to benchmark the hPSC-derivatives to control samples from FL and AL for assessment of their maturation, a public dataset generated with the Illumina platform were downloaded and merged with our Affymetrix dataset. Expectedly, a strong batch effect was observed due to the issue of different technical platforms, which are known to have large affects on the gene expression measurements. Thus, the known putative batch effects were *different technical platforms* and *differences in time point* for the performance of the experiments.

2.3 Merging of datasets

The merging of the cardiac biopsy dataset was performed on probe set ID since identical arrays were used to generate both the input datasets. In total 33,297 probe sets were included in the merged dataset. For the hepatic differentiation study the merging of datasets was

performed on gene symbol since that was the common identifier across the two datasets. In total of 12,921 unique gene symbols were common across both the input datasets in this study design.

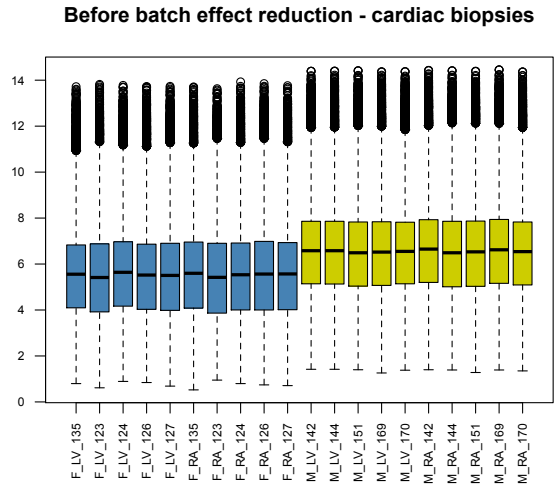


Figure 1: Boxplot of the expression range of the arrays in the merged dataset from the cardiac biopsy study before batch effect reduction. Blue boxes represents arrays from the male patients and yellow boxes represent data from the female patients.

2.4 QC analysis of the merged datasets

The expression range, the distribution of data, and the correlation between samples in the merged datasets were explored using boxplots and hierarchical clustering, and significant batch effects were observed in both the cardiac biopsy dataset and in the hepatic differentiation dataset. Fig. 1 and Fig. 2 illustrates identified differences between the two batches in the cardiac biopsy study and in Fig. 3 and Fig. 4 differences between the two different array platforms in the hepatic differentiation study design are shown.

2.5 Batch effect reduction using ComBat

To facilitate modeling of the batch effect two overlapping samples across both batches were used in each of the study designs. In the cardiac biopsy study two identical RNA samples were added to the global transcriptional experiments and used as overlapping samples when applying the ComBat algorithm. The batch factor specified as input to the algorithm was the two different transcriptional experiments, one including cardiac biopsies from male patients and the other one including cardiac biopsies from female patients. For the hepatic differentiation study no overlapping RNA samples were available but samples from AL were available in both the input datasets, and although these samples were not from identical biological material they still represented similar tissue material. Thus, the two AL control samples in the differentiation dataset were defined as overlapping samples and used by the ComBat algorithm for modeling of batch differences. The batch factor specified as input to the algorithm was difference of technical platforms.

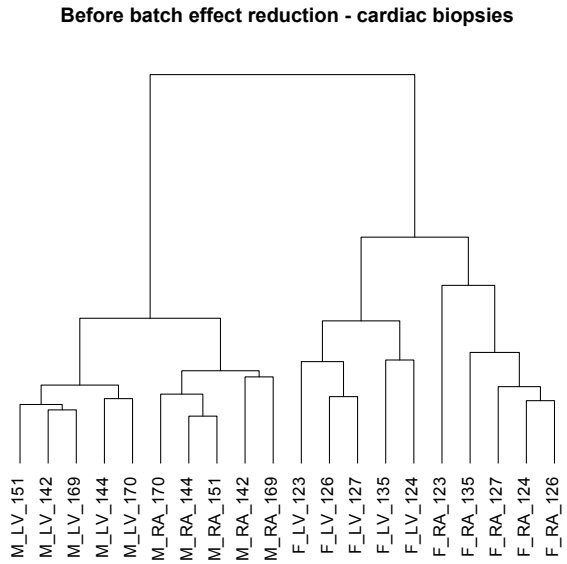


Figure 2: Hierarchical clustering before reduction of the batch effect. The clustering shows two distinct clusters reflecting that the time point of the experiment represents the main transcriptional difference in this merged dataset.

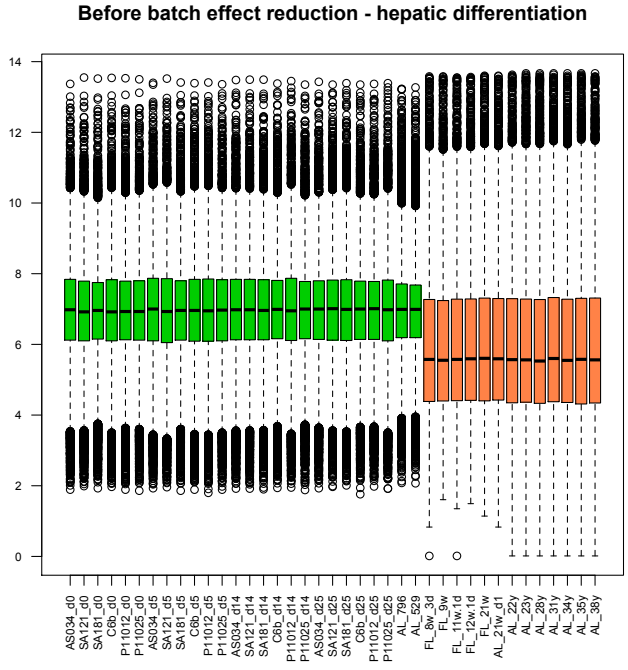


Figure 3. Boxplot showing the expression range of the arrays in the merged dataset from the hepatic differentiation study before the batch effect reduction. Green boxes represents arrays from the Affymetrix platform and orange boxes represent data generated with the Illumina platform. Notice the big platform related differences in the first quartile of the data.

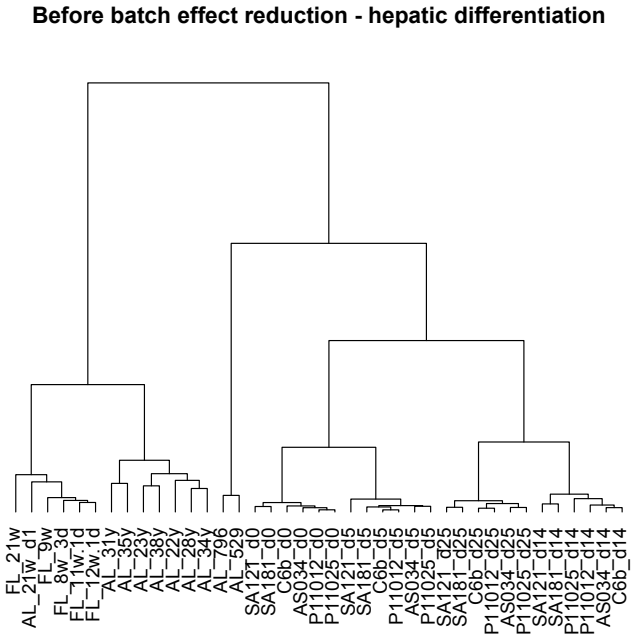


Figure 4. Hierarchical clustering before reduction of the batch effect. The dendrogram shows two distinct clusters reflecting that experimental platform is the main difference in this merged dataset since the two AL samples (AL_796 and AL_529) that where run on the Affymetrix platform clustered with the stem cell derivatives instead of with the other AL samples.

3. Results

Results from this study demonstrated that the ComBat algorithm managed to reduce the batch effects present in the two study designs explored in this work. Both these study designs contained merged large-scale omics data, and represent two different sources of batch effects that commonly are faced in various types of omics data analysis. In the first example the data generation were divided into two experimental runs, with the risk of introduction of batch effects in the merged dataset. The second situation represents batch effects that source from adding extra samples needed in subsequent downstream analysis. In our case the purpose with adding data was to add specific control samples to a present dataset by utilizing data from a public database.

3.1 Results from the cardiac biopsy dataset

The boxplots and the hierarchical clustering after applying the ComBat algorithm to the merged cardiac biopsy dataset show that the main proportions of batch effects successfully were eliminated and the merged datasets demonstrate similar distribution of the expression range across all arrays. As shown in the boxplot in Fig 5 the distribution of the expression values are now at similar range for all the included arrays in this merged dataset. Notably, also the data points in the first quartile and in the fourth quartile (those data points that fall outside the boxes) show highly similar expression range across all the arrays. And the median value for each array harmonized across the merged experiment. A hierarchical clustering of the adjusted data showed that the reduction of batch effects did not eliminate true biological differences since the known differences between the LV and the RA are still preserved in the data. This is demonstrated by the grouping of samples according to tissue location as shown in Fig. 6.

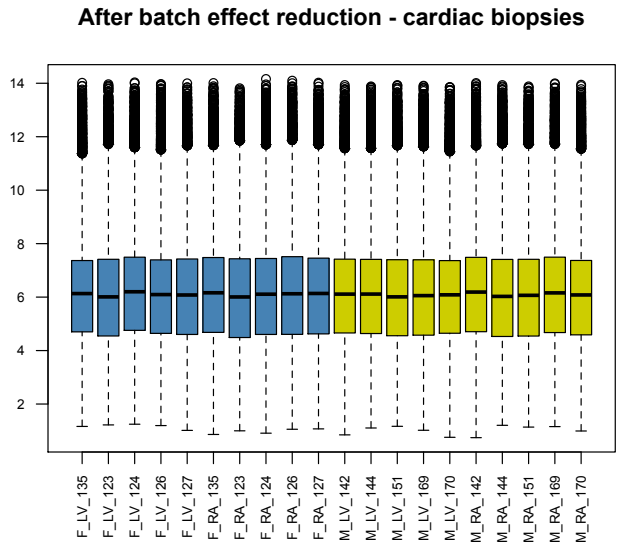


Figure 5. Boxplot showing the expression range of all the arrays in the merged dataset of cardiac biopsies. The major visible batch effects are now removed from the dataset.

3.2 Results from the hepatic differentiation dataset

Also for the hepatic differentiation dataset the elimination of batch effects was successful although the variation of the values in the fourth quartile was slightly higher for the FL samples as shown in the boxplot in Fig 7. A putative explanation is that this may reflect a true biological variation that perhaps can be addressed to the different developmental weeks of the embryo from which the FL samples were derived. After reduction of the batch effect between the technical platforms an biologically relevant grouping from the hierarchical clustering was achieved which showed a perfect classification of the different groups of samples in the merged dataset as illustrated in Fig 8.

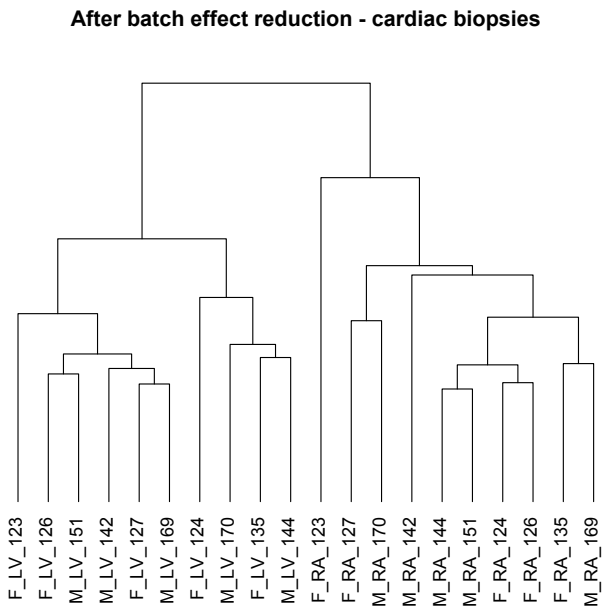


Figure 6. Hierarchical clustering after batch effect reduction resulted in more biologically relevant groups in the cardiac biopsy dataset. Two distinct clusters based on localization of the analysed biopsies rather than the gender were generated.

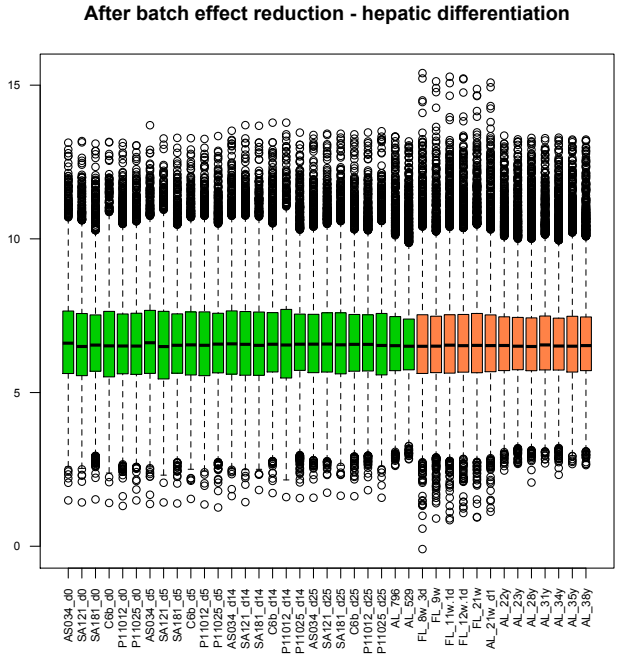


Figure 7. Boxplot showing the expression range of all the arrays in the merged dataset of hepatic differentiation, including the samples from FL and AL for benchmarking.

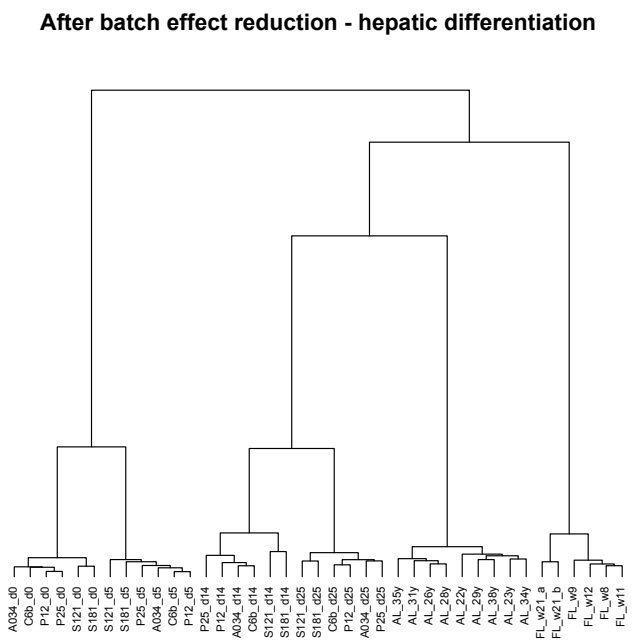


Figure 8: Hierarchical clustering of samples in the hepatic differentiation study after successful reduction of the batch effect introduced in the merged dataset. Perfectly biologically relevant grouping was achieved after the batch effect adjustment.

4. Discussion and conclusion

Huge quantities of global omics datasets are now routinely generated and the amount of available biomedical big data is increasing exponentially. If utilized in an optimal way the availability of these vast data provides an incredible resource that will likely revolutionize the area of medical research. Moreover, the increasing number of data repositories for storage of biomedical big data greatly facilitates various types of meta-studies, in which many different datasets can be merged and analyzed in combination. However, combining data from multiple experiments and sources, generated using different platforms and molecular techniques are not trivial and increases the risk to introduce batch effects in the merged datasets as well. Putative batch effects that may have been introduced during data integration processes needs to be identified and corrected for before performance of the subsequent downstream data analysis. However, there is always a risk that during elimination of batch effects, true biological variation is also mistakenly removed. Thus, more investigations are needed to evaluate and compare different approaches for reducing batch effects introduced during data integration.

In this study the ComBat algorithm has been applied on two types of study designs representing two common batch effect issues in omics data analysis. Results from

our work demonstrated that the ComBat algorithm could successfully reduce the observed batch effects in both the merged datasets while at the same time preserving the true biological variation. The importance of correction for batch effects before continuing with the downstream analysis has also been emphasized by the results from the hierarchical clustering analysis, where improved biologically relevant groupings were achieved after the batch effect reduction was performed.

In conclusion, the results from this study demonstrate the applicability of the ComBat method to correct for various types of batch effects introduced during merging of large-scale omics data. Through this work the importance of exploring expression range and data distribution after merging of large–scale datasets has been highlighted.

Acknowledgments

This work was supported by the University of Skövde, Sweden, under grants from the Knowledge Foundation [2014/0301].

5. References

[1] W. W. B. Goh, W. Wang, and L. Wong, "Why Batch Effects Matter in Omics Data, and How to Avoid Them," *Trends Biotechnol,* vol. 35, pp. 498-507, Jun 2017.

[2] T. Han, C. D. Melvin, L. Shi, W. S. Branham, C. L. Moland, P. S. Pine*, et al.*, "Improvement in the reproducibility and accuracy of DNA microarray quantification by optimizing hybridization conditions," *BMC Bioinformatics,* vol. 7 Suppl 2, p. S17, Sep 06 2006.

[3] M. J. Larsen, M. Thomassen, Q. Tan, K. P. Sorensen, and T. A. Kruse, "Microarray-based RNA profiling of breast cancer: batch effect removal improves cross-platform consistency," *Biomed Res Int,* vol. 2014, p. 651751, 2014.

[4] N. Kolesnikov, E. Hastings, M. Keays, O. Melnichuk, Y. A. Tang, E. Williams*, et al.*, "ArrayExpress update--simplifying data submissions," *Nucleic Acids Res,* vol. 43, pp. D1113-6, Jan 2015.

[5] R. Edgar, M. Domrachev, and A. E. Lash, "Gene Expression Omnibus: NCBI gene expression and hybridization array data repository," *Nucleic Acids Res,* vol. 30, pp. 207-10, Jan 01 2002.

[6] B. Ulfenborg, A. Karlsson, M. Riveiro, C. Ameen, K. Akesson, C. X. Andersson*, et al.*, "A data analysis framework for biomedical big data: Application on mesoderm differentiation of human pluripotent stem cells," *PLoS One,* vol. 12, p. e0179613, 2017.

[7] D. Gomez-Cabrero, I. Abugessaisa, D. Maier, A. Teschendorff, M. Merkenschlager, A. Gisel*, et al.*, "Data integration in the era of omics: current and future challenges," *BMC Syst Biol,* vol. 8 Suppl 2, p. I1, 2014.

[8] W. E. Johnson, C. Li, and A. Rabinovic, "Adjusting batch effects in microarray expression data using empirical Bayes methods," *Biostatistics,* vol. 8, pp. 118-27, Jan 2007.

[9] J. T. Leek and J. D. Storey, "Capturing heterogeneity in gene expression studies by surrogate variable analysis," *PLoS Genet,* vol. 3, pp. 1724-35, Sep 2007.

[10] J. A. Gagnon-Bartsch and T. P. Speed, "Using control genes to correct for unwanted variation in microarray data," *Biostatistics,* vol. 13, pp. 539-52, Jul 2012.

[11] Y. Hasin, M. Seldin, and A. Lusis, "Multi-omics approaches to disease," *Genome Biol,* vol. 18, p. 83, May 05 2017.

[12] S. Huang, K. Chaudhary, and L. X. Garmire, "More Is Better: Recent Progress in Multi-Omics Data Integration Methods," *Front Genet,* vol. 8, p. 84, 2017.

[13] J. Asp, J. Synnergren, M. Jonsson, G. Dellgren, and A. Jeppsson, "Comparison of human cardiac gene expression profiles in paired samples of right atrium and left ventricle collected in vivo," *Physiol Genomics,* vol. 44, pp. 89-98, Jan 18 2012.

[14] N. Ghosheh, B. Kuppers-Munther, A. Asplund, J. Edsbagge, B. Ulfenborg, T. B. Andersson*, et al.*, "Comparative transcriptomics of hepatic differentiation of human pluripotent stem cells and adult human liver tissue," *Physiol Genomics,* vol. 49, pp. 430-446, Aug 1 2017.

[15] N. Ghosheh, B. Olsson, J. Edsbagge, B. Kuppers-Munther, M. Van Giezen, A. Asplund*, et al.*, "Highly Synchronized Expression of Lineage-Specific Genes during In Vitro Hepatic Differentiation of Human Pluripotent Stem Cell Lines," *Stem Cells Int,* vol. 2016, p. 8648356, 2016.

Predicting Pathways from Untargeted Metabolomics Data

Daniel Salinas and Brendan Mumey
Gianforte School of Computing
Montana State University
Bozeman, MT, USA
{daniel.salinas@msu, mumey@cs}.montana.edu

Ronald K. June
Mechanical & Industrial Engineering
Cell Biology & Neuroscience
Montana State University
Bozeman, MT, USA
rjune@montana.edu

Abstract

We propose an approach for predicting active pathways from untargeted metabolomics data by minimizing the number of pathways needed to fully explain the features of the data. The approach was tested on data taken from cells infected with yellow fever virus and compared with alternative approaches from literature. Our methodology yielded predictions that were validated by data separate from metabolomics and were a more complete description of the infection phenotype. We also introduce an alternative formulation that would allow leveraging the retention time information provided by liquid chromatography-mass specrometry.

1 Introduction

Untargeted metabolomics is a useful tool in providing an unbiased representation of cell metabolism. However, to match current metabolic models, spectra resulting from metabolomics must be mapped to known metabolites. These metabolites yield insight into which metabolic pathways are active and indicate the functional state of the cell.

Obtaining metabolite identities from spectra has become a bottleneck in metabolomics studies, requiring further experiments to verify metabolite identities before pathway activities can be deduced. There can be thousands of features that require identification, creating the need for approaches that can automate either active metabolic pathway inference or metabolite identification.

2 Related Work

We give a brief overview of pathway enrichment analysis from metabolomics data. Assessing enrichment directly, i.e. without mapping the spectral features to metabolites prior to analysis, is a relatively recent development. Previously, enrichment analyses had been developed for other "-omics" data or required metabolites rather than spectra as input. An overview of these is given in [12].

To the best of our knowledge, *mummichog* [7] has become the standard approach to pathway enrichment analysis from LC-MS (liquid chromatography coupled to mass spectrometry) data. First, LC-MS data is obtained using a wide range of mass-to-charge ratios to capture as many features as possible. A subset of those features is then identified as differing significantly between the experimental and control groups. The authors of [7] denote this subset *Lsig*, and the full feature set *Lref*. Samples from *Lref* are then used to estimate the likelihood that a pathway will contain features if the samples are of size $|Lsig|$. Over many samples, a distribution is generated that can then be used to gauge which pathways have an abnormally large intersection with *Lsig*. These are the enriched pathways.

A recent approach, PIUMet, infers metabolites from untargeted LC-MS spectra using a prize-collecting Steiner tree [6]. This is an optimization problem on a graph, where the solver must select a set of edges to form trees that connect prize nodes while minimizing the total edge cost. From an input of LC-MS features, PIUMet defines the problem by letting the prize nodes of the graph be the m/z values of features, connecting them via a network of proteins and metabolites. Specifically, there is are nodes for each metabolite and protein, and edges between a protein-protein or protein-metabolite pair exist if there is evidence they interact. Edge costs represent the confidence of the interaction, with a higher cost representing a lower confidence. The optimal set of trees with respect to cost and penalty represent active pathways. We refer the reader to [9] for further details. We briefly discuss PIUMet in §7.

3 Problem Definition

Inferring active pathways from untargeted metabolomics requires reconciling the LC-MS data, consisting of a set of mass-to-charge ratio and retention time pairs, to the model, a collection metabolites connected by reactions in pathways. Metabolites must, either implicitly or explicitly, be matched with mass-to-charge ratio (also denoted m/z value) and retention time pairs. Our approach matches the metabolites implicitly, by selecting the minimal set of

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

pathways that contain enough metabolites with the observed mass-to-charge ratios to represent the different retention times observed.

We will refer to this problem as the *Minimum Pathway Cover* problem and abbreviate it as MPC. We define MPC in terms of pathways, LC-MS data and retention times. Here we note that MPC is an instance of a more general problem that can be adapted to leverage data about metabolite properties if it is available. We explain this problem and its applications in §9.

To define MPC, let

- $\mathbf{Z}$ be the set of observed m/z values, and
- $\mathbf{R}$ be the set of observed retention times.

The LC-MS data is therefore a subset $X \subseteq \mathbf{Z} \times \mathbf{R}$. Note that there is often a tolerance associated with m/z values; in enrichment analyses it is common to consider m/z values that differ by less than 10 p.p.m. the same value. In this case, $\mathbf{Z}$ would be a set of m/z value intervals. More details are given in §6. The model consists of

- a set $\mathbf{M}$ of metabolites,
- a mapping $m : \mathbf{M} \to \mathbf{Z}$, where $m(t_j) = z_i$ iff metabolite t_j has m/z value z_i, and
- a collection of pathways $P \subseteq 2^{\mathbf{M}}$, where $2^{\mathbf{M}}$ is the powerset of $\mathbf{M}$.

A set of pathways $C \subseteq P$ is capable of generating X if

$$\bigcup_{p \in C} \{m(t_j) \mid t_j \in p\} = \mathbf{Z}.$$

MPC finds the smallest C that generates X with enough metabolites to cover (see Fig. 1) the distinct retention times. Let $c(z_i)$ be the number of distinct retention times observed to occur with z_i in the data,

$$c(z_i) = \big|\{r_k \mid (z_i, r_k) \in X\}\big|.$$

Given $C \subseteq P$, the metabolites matched to z_i by C are

$$\bigcup_{p \in C} m^{-1}(z_i) \cap p.$$

For all $z_i \in \mathbf{Z}$, the number of metabolites matched to z_i must be at least the number of distinct retention times for z_i:

$$c(z_i) \leq \left| \bigcup_{p \in C} m^{-1}(z_i) \cap p \right|.$$

4 Implementation

MPC can be encoded as an integer linear program (ILP). The variables of the ILP are defined by letting

- x be a binary vector of length $|P|$, and
- for each $z_i \in \mathbf{Z}$, c_i be a binary vector of length $|\mathbf{M}|$.

We use x and the collection of c_i to encode a valid solution C. We assign each $p \in P$ an index n and let x be a binary representation of C, where $x_n = 1$ if $p_n \in C$ and 0 otherwise. We assign each metabolite $t \in \mathbf{M}$ an index j and let $c_{ij} = 1$ if $t_j \in p_n$, $p_n \in C$, and $m(t_j) = z_i$. We let $c_{ij} = 0$ otherwise.

Given these variables, the ILP representation of MPC is:

$$\min_x \ ||x||_1 \tag{1}$$

$$c_{ij} \leq \sum_{\{n \mid p_n \in P, t_j \in p_n\}} x_n \qquad \forall i, j \tag{2}$$

$$c(z_i) \leq \sum_{\{j \mid t_j \in \mathbf{M}\}} c_{ij} \qquad \forall i \tag{3}$$

$$x_n \leq c_{ij} \qquad \text{if } t_j \in p_n, m(z_i) = t_j \tag{4}$$

(1) sets as the objective the minimizing of the number of pathways in C by minimizing the sum of the x_n. Since x_n is a binary vector, this is equivalent to minimizing the number of x_n with value 1. Constraint (2) ensures that $c_{ij} = 0$ unless some pathway p_n that contains metabolite t_j has been chosen as part of C. Note that all c_{ij} are binary variables, so even if multiple pathways in C assign metabolite t_j to z_i the value of c_{ij} is at most 1 in that case; it is 0 if no such pathways are part of C. Constraint (3) sets the lower bound on the number of metabolites assigned to z_i to the number of distinct retention times observed for that m/z value. Constraint (4) ensures that if pathway p_n contains a metabolite t_j that has m/z value z_i, then whenever pathway $p_n \in C$ (i.e. whenever $x_n = 1$), metabolite t_j is assigned to z_i (i.e. $c_{ij} = 1$). The ILP was solved using CPLEX [5].

5 Experiments

We tested our approach on previously published data obtained from monocyte-derived dendritic cells (moDC) stimulated by yellow fever virus (YF-17D vaccination strain). This data was chosen because (i) the presence of key metabolites was verified by tandem mass spectrometry, and (ii) gene expression analysis with direct measurement confirmed the occurrence of phenotypes associated with YF-17D infection, and (iii) it was analyzed in-depth with *mummichog* by its authors [7].

The moDCs were infected, mock-infected, or used as baseline controls. Metabolome samples were taken at 0,

6, and 24 hours. LC-MS was full-scan (m/z between 85 and 2000) using a reverse phase C18 column, and samples were ionized using positive electrospray ionization. [11, 7] Tandem mass spectrometry was performed using LTQ-FTMS. Features associated with infection were identified by comparing the intensities of features in infected samples at 6 h. to intensities of those features in both mock-infected samples (also 6 hours post mock infection) and baseline controls (0 hours) and selecting those that differed significantly. Those features significantly different (according to Student's t-test) between the infected samples and both the baseline controls and the mock-infected samples were selected for pathway enrichment analysis. A comprehensive description of significant features detection, from the statistical methods used to the LC-MS extraction, can be found in [7].

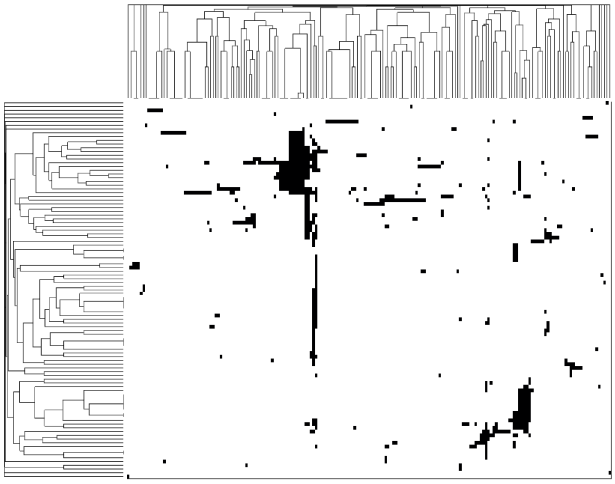


Figure 1: Heatmap illustrating the m/z each pathway covers. Rows correspond to patwhways and columns to m/z values. If a pathway can cover an m/z value that square is black, and white otherwise. Rows and columns clustered by similarity.

Experiments were run using the pathways in the model used in [7] to prevent the comparison of MPC to *mummichog* from being affected by model discrepancies. A heatmap showing the coverage of each pathway to the m/z values in X is shown in 1 As mentioned, X was chosen to be the features deemed significantly different between the infected and non-infected groups. At $p = 0.05$, this consists of 601 features (m/z value and retention time pairs). Pathways were obtained using the "human_model_mfn" model provided with version 1.0.5 of *mummichog*. Also provided with the software are a mapping from metabolites to masses and a function that generates m/z values resulting from the ionization process from a base mass. These were used to define $m : \mathbf{M} \to \mathbf{Z}$. The mapping $c : \mathbf{Z} \to \mathbb{N}$ was defined using X as described in §4, except in those cases where

$$\left|m^{-1}(z_i)\right| < \left|\{r_k \mid (z_i, r_k) \in X\}\right|,$$

i.e. there were more retention times associated with z_i in the data than metabolites in the model that could map to z_i. In these cases, $|m^{-1}(z_i)|$ was used as the lower bound $c(z_i)$.

6 Results

MCP identified a total of 43 active pathways. Infection with YF-17D is known to induce cellular stress response, namely, interleukin cytokines (IL)-12p40, IL-6, and interferon-α via toll-like receptors (TLRs) 2, 7 , 8 and 9. [10] Viral infection may trigger the production of nitric oxide from arginine precursors as a defense mechanism, activated via TLRs. [8] Infection also causes nucleotide synthesis as part of viral replication. Confirmation of these processes was obtained from measurements independent of metabolomics [7]. The presence of IL-6 was confirmed via direct measurement. Negative feedback of nitric oxide synthesis was detected by direct measurement. Downregulation of glutathione synthesis was detected by transcriptomics. Tandem mass spectrometry confirmed the presence of, among others, arginine, citrulline, AMP, GMP, glutamate, xanthine, inosine, glutathione, GMP, and GSSG.

These measurements confirm the correctness of pathways in C. For example, the "Aspartate and arparagine metabolism" pathway covers m/z values corresponding to arginine, citrulline, AMP, glutathione, and GSSG; this pathway, along "Purine metabolism" (covers m/z values of inosine and xanthine) contain a network of reactions that synthesize, from arginine, glutathione and purine precursors. Additionally, citrulline is a by-product of nitric oxide synthesis. While both of these pathways are in *mummichog*'s enriched set, MCP allowed for the inclusion of pathways ancillary, but important, to these processes. For example, including the pentose phosphate pathway, as it is in MCP, gives a likely explanation of how fatty acid synthesis factor NADPH is regenerated. [3] Fatty acid synthesis is an important response to infection in dendritic cells, as cytokine production demands a larger endoplasmic reticulum and a larger Golgi apparatus. [4] Indeed, many other the pathways found by MCP were parts of fatty acid metabolism.

A comparison summary is given in Table 1. The statistics used for comparison come from [7] and accompanying supplementary material Dataset S1 and Table S1. Solving MPC resulted in 43 pathways identified as active, in contrast to *mummichog*'s 21. Of these, 14 were common to both approaches. The greater number of pathways generated by MPC is expected. MPC requires, via lower bounds $c(z_i)$, that all features that can be explained by the model be explained by choosing a relevant pathway. However, *mummichog*'s more conventional enrichment analysis identifies as active pathways only those that have a statistically

anomalous presence of $^m/_z$ values in the significantly different feature list. The full set of 7995 features observed in all samples is used to derive an empirical distribution on the presence of features in each pathway. This set of features, denoted *Lref*, contains the set of 601 significantly different features, referred to as *Lsig*. The distribution is generated by selecting samples of 601 features from *Lref*. A variant of Fisher's exact test is then used to, for each pathway, record a p value corresponding to the likelihood that this pathway has the observed overlaps with *Lsig* and the sample from *Lref* under the null hypothesis that there is no relationship between a metabolite being present in the pathway and whether it belongs to *Lsig* or the sample from *Lref*. This method is adapted from [2]. Over multiple samples from *Lref*, a distribution of p values is observed. Those pathways with p values below some threshold are deemed active. This distribution is necessary to detect significant pathways. A study we ran (not shown for conciseness) using Fisher's exact test with random metabolite assignments from possible biological candidates to $^m/_z$ showed low significance for all pathways (pathways had many elements from *Lref* even if they had elements from *Lsig*). We refer the reader to [7] for more details.

Table 1: Approach Summary

	mummichog	MPC
Total pathways	21	43
Unique pathways	7	29
$^m/_z$ matched	132	188
$^m/_z$ matched per pathway (avg.)	6.29	4.37
Metabolites per feature (avg.)	1.71	1.34

Some features in *Lsig* may therefore be left unexplained by the pathways selected by *mummichog*. Specifically, these are features corresponding to metabolites found only in those pathways that relate to *Lsig* and *Lref* to similar degree, i.e. the features that correspond to metabolites only found in pathways whose p value is not low enough. This was observed empirically in the number of features matched. In Dataset S1, 330 $^m/_z$ values are given metabolite predictions based on the pathways selected. Of these 330 values, 138 belong to features in *Lsig*. The number of features matched by MPC is the sum of the lower bounds $c(z_i)$, in this case 192; these features have 188 distinct $^m/_z$ values. Note that this is not 601, as might be expected, since $c(z_i)$ is often zero: there are no metabolites in the model that correspond to the given mass. Enforcing full coverage of features where possible, MPC will tend to select more pathways than *mummichog* and match more features.

We also observed that, on average, fewer metabolites were assigned per mass by solving MPC when compared to *mummichog*. Comparing the number of metabolites assigned per $^m/_z$ value shows that, though the distributions are similar,

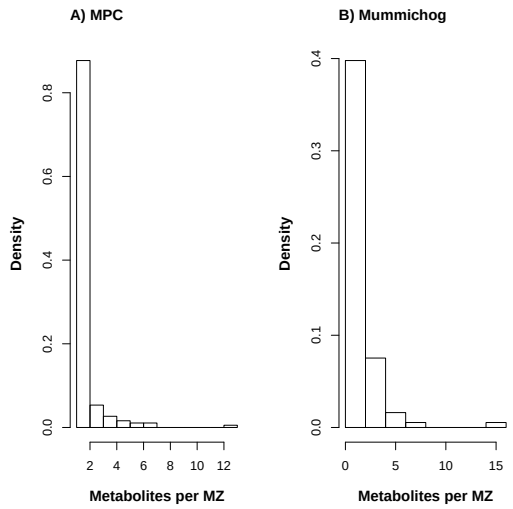


Figure 2: Distribution of the number of metabolites assigned to $^m/_z$ values for A) MPC and B) *mummichog*. MPC tends to assign less overall.

the fraction of $^m/_z$ that match to a single metabolite is higher for MPC than for than for *mummichog*. (Fig. 2) This is expected if we consider that $c(z_i) = 1$ in the majority of the cases. This means that there was a single feature observed per $^m/_z$ value and so MPC only requires a single metabolite assignment to be satisfied. Interestingly, the majority of these one-to-one masses are $^m/_z$ values that are contained in pathways unique to MPC. When considering only those $^m/_z$ values that are common to both approaches, the distributions are more similar. Since pathways are considered enriched only if they have a relatively high number of metabolites from *Lsig*, *mummichog* will tend to explain only those $^m/_z$ values that occur in groups. Enrichment also has the implication that if a group of metabolites from *Lsig* tends to repeat throughout the pathways in the model, the pathways that contain them will all be reported as active. This can be observed when comparing the average overlap within pathways in MCP (Fig. 3) to that within the enriched pathways (Fig. 4). Pathways in *mummichog* tend to overlap more with each other, both when comparing masses explained and metabolites in those pathways.

7 PIUMet

We also include PIUMet results with some caveats. First, we limit our discussion to $^m/_z$ value matching to metabolites. PIUMet is a pathway inference, rather than pathway enrichment, tool. As such, the pathways returned are networks of proteins and metabolites and may not correspond to known catabolic or anabolic pathways or may match pathways ambiguously. Second, the algorithm was run from the PIUMet website[1], forcing the use of the PPMI

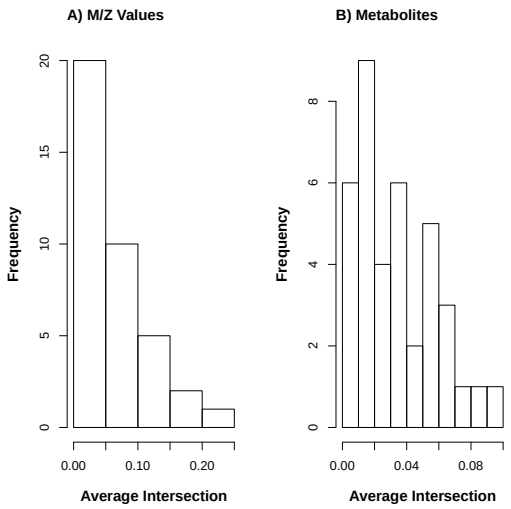


Figure 3: Average size of the intersection of a pathway selected by MCP with another selected by MCP, where a pathway is either A) a set of $^m/_z$ values or B) a set of metabolites.

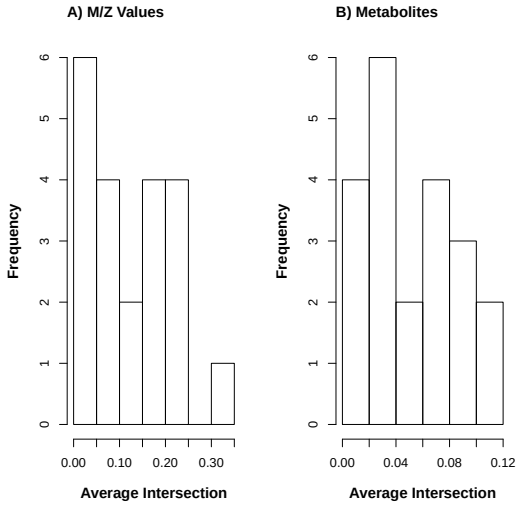


Figure 4: Average size of the intersection of a pathway selected by MCP with another selected by *mummichog*, where a pathway is either A) a set of $^m/_z$ values or B) a set of metabolites.

network, a different model than the one used throughout the rest of this paper.

The prize w_i of each feature z_i in *Lsig* was defined as

$$w_i = -\log p_i, \tag{5}$$

where p_i is the p value obtained from the t-test measuring the significance of the difference in z_i between the infected cells and the other two groups. This definition of w_i is used in [9]. All other parameters were left as the defaults provided by the website.

The metabolites inferred by PIUMet and confirmed by tandem mass spectrometry include GSSG, citrulline, and xanthine, PIUMet output includes a top metabolite match for each $^m/_z$ value. Of these 103 metabolites, 36 were also inferred by MCP. These include metabolites in pathways shared by all approaches (e.g. Methionine and cysteine metabolism, Pyrimidine metabolism). Overall, PIUMet matched more $^m/_z$ values (257) to metabolites than MCP or *mummichog*, though this is likely due to its use of a larger model (114,100 vs. 3,565). Considering all possible (927) matches returned by PIUMet emphasizes this difference as each $^m/_z$ value is assigned 3.6 metabolites on average, compared to the 1.71 and 1.34 by *mummichog* and MCP, respectively. Despite the much larger number of possible matches, the intersection with the metabolites assigned by MCP increases only to 62. We conclude that PIUMet shows its ability to formulate de novo pathways, while agreeing with established literature.

8 Conclusions

We have shown that choosing the smallest set of pathways capable of producing the spectral features observed in untargeted metabolomics data from cells infected with YF-17D yields pathways consistent with known phenotypes of infection. We introduced this problem as *Minimum Pathway Cover* and show it can be solved by solving an integer linear program using CPLEX, a popular optimization suite. Setting a requirement for full coverage explains those features that would be ignored by enrichment analyses, and requiring a minimal set of pathways reduces the ambiguity in the results by selecting fewer metabolites per feature observed. These potentially ignored features also related metabolic pathways known to be active during infection. These initial results encourage further investigation into which combinations of data and metabolic network topologies will cause either MCP or enrichment analyses to be more effective.

9 Future Work

To allow comparison to *mummichog*, we have considered selecting a cover as an assignment of metabolites to $^m/_z$ values. However, setting lower bounds $c(z_i)$ allows for a mapping of metabolites to features. For example, if $c(z_i) = 2$, then two features $(z_i, r_1), (z_i, r_2)$ with distinct retention times $r_1 \neq r_2$ were observed in the data set X. Then, knowing C has pathways with at least two metabolites with $^m/_z$ value z_i, an attempt may be made to map metabolites to either (z_i, r_1) or (z_i, r_2).

However, metabolites with similar chemical properties will elute at the same time, meaning that multiple metabo-

lites may correspond not only to the same m/z value but to the same (z_i, r_k) pair. In these cases, even though more than $c(z_i)$ metabolites may be mapped to $c(z_i)$, not all the r_k have been covered. However, with an alternative definition of MCP we may address the problem.

MCP may be defined in generic terms. Since this is a variant of the Set Cover problem, we will denote this generic version *Color Set Cover* and abbreviate it CSC. We let $\mathbf{Z}, \mathbf{R}, \mathbf{M}$ be generic sets and $P \in 2^{\mathbf{M}}$ be a set of subsets of $\mathbf{M}$. Let $X \subseteq \mathbf{Z} \times \mathbf{R}$ be the problem input. In terms of Set Cover, X is the universe and P is the family of sets a cover will be selected from. In ordinary Set Cover, $P \in 2^{\mathbf{Z}}$ and a cover $C \subseteq P$ is selected such that

$$\bigcup_{p \in C} p = \{z_i \mid (z_i, r_k) \in X\} = \mathbf{Z}.$$

In CSC, the mapping $m : \mathbf{M} \to \mathbf{Z}$ determines whether a set $p \in P$ covers a value in X. As in §3, p covers $z_i \in \mathbf{Z}$ if p contains an element $t_j \in \mathbf{M}$ such that $m(t_j) = z_i$. We denote $\mathbf{M}$ as the set of colors. Choosing a cover therefore implicitly assigns a set of colors to each z_i. Moreover, CSC imposes a further constraint on C, namely that the number of colors assigned to each to each z_i be greater than or equal to the number of r_k paired with it in X. We have, for MCP, chosen the sets $\mathbf{Z}$, $\mathbf{R}$, and $\mathbf{M}$ as described in §3. To address the problem of multiple metabolites having the same retention time, we let $\mathbf{Z}$ be the set of m/z values as before, but change $\mathbf{R}$ and $\mathbf{M}$. We let $\mathbf{R}$ be retention time classes and $\mathbf{M}$ classes of metabolites, both determined by the LC-MS method. For example, HILIC columns separate metabolites according to polarity. Then, we can label each metabolite with an element $t_j \in \mathbf{M}$, where metabolites have the same label t_j if their polarity is very similar. Set $\mathbf{R}$ could be the set of retention times, or it could be a set of non-overlapping retention time ranges observed in the data (e.g. $\mathbf{R}$ = { 100-120, 50-70}). While it may be intuitive to let $\mathbf{R} = \mathbf{M}$, letting them be different allows us to select the appropriate number of metabolites with distinct retention times while not assigning them to any particular range of retention times, since the retention time of a metabolite is very difficult to predict a priori. Once a cover is obtained, the metabolite classes may be sorted according to polarity and a tentative feature matching be obtained. We plan to implement this strategy in future studies.

Acknowledgment: This work was funded by NSF-CMMI 1554708 and NSF-ABI 1542262.

References

[1] Fraenkel lab – piumet. `http://fraenkel-nsf.csbi.mit.edu/piumet2/`. Accessed: 2018-01-18.

[2] G. F. Berriz, O. D. King, B. Bryant, C. Sander, and F. P. Roth. Characterizing gene sets with funcassociate. *Bioinformatics*, 19(18):2502–2504, 2003.

[3] M. Cortese, C. Sinclair, and B. Pulendran. Translating glycolytic metabolism to innate immunity in dendritic cells. *Cell metabolism*, 19(5):737–739, 2014.

[4] B. Everts, E. Amiel, S. C.-C. Huang, A. M. Smith, C.-H. Chang, W. Y. Lam, V. Redmann, T. C. Freitas, J. Blagih, G. J. Van Der Windt, et al. Tlr-driven early glycolytic reprogramming via the kinases tbk1-ikk [epsiv] supports the anabolic demands of dendritic cell activation. *Nature immunology*, 15(4):323–332, 2014.

[5] IBM. *ILOG CPLEX Optimization Studio*, 2017.

[6] D. S. Johnson, M. Minkoff, and S. Phillips. The prize collecting steiner tree problem: theory and practice. In *SODA*, volume 1, page 4, 2000.

[7] S. Li, Y. Park, S. Duraisingham, F. H. Strobel, N. Khan, Q. A. Soltow, D. P. Jones, and B. Pulendran. Predicting network activity from high throughput metabolomics. *PLoS computational biology*, 9(7):e1003123, 2013.

[8] S. M. Morris Jr. Arginine: master and commander in innate immune responses. *Sci Signal*, 3(135):e27, 2010.

[9] L. Pirhaji, P. Milani, M. Leidl, T. Curran, J. Avila-Pacheco, C. B. Clish, F. M. White, A. Saghatelian, and E. Fraenkel. Revealing disease-associated pathways by network integration of untargeted metabolomics. *Nature methods*, 13(9):770–776, 2016.

[10] T. Querec, S. Bennouna, S. Alkan, Y. Laouar, K. Gorden, R. Flavell, S. Akira, R. Ahmed, and B. Pulendran. Yellow fever vaccine yf-17d activates multiple dendritic cell subsets via tlr2, 7, 8, and 9 to stimulate polyvalent immunity. *Journal of Experimental Medicine*, 203(2):413–424, 2006.

[11] Q. A. Soltow, F. H. Strobel, K. G. Mansfield, L. Wachtman, Y. Park, and D. P. Jones. High-performance metabolic profiling with dual chromatography-fourier-transform mass spectrometry (dc-ftms) for study of the exposome. *Metabolomics*, 9(1):132–143, 2013.

[12] J. Xia and D. S. Wishart. Metpa: a web-based metabolomics tool for pathway analysis and visualization. *Bioinformatics*, 26(18):2342–2344, 2010.

Automated Biomedical Text Classification with Research Domain Criteria

Mohammad Anani and Indika Kahanda
Gianforte School of Computing, Montana State University
Bozeman, Montana, 59717, USA
mohammad.anani@student.montana.edu
indika.kahanda@montana.edu

Abstract

Research Domain Criteria (RDoC) is a recently introduced framework for accurate diagnosis of mental illness. Developing a method to automate the process of labeling biomedical articles with RDoC constructs is highly useful to advance research efforts in the area of mental illness. Therefore, this study focuses on exploring the feasibility of developing a tool for this purpose. Using a gold-standard dataset of about 40,000 Medline abstracts annotated with 26 RDoC constructs, we model this both as a binary and a multilabel classification problem, to perform document classification using several supervised learning algorithms. We use a simple bag-of-words model with standard preprocessing steps such as stemming and stop words removal. According to a performance evaluation obtained through 5-fold cross validation, we observe that overall, multilabel Artificial Neural Networks classifier performs best with an excellent average AUROC of 96% across all the constructs. Interestingly, all the binary classifiers also show a very high-level of performance. However, the cohort of binary classifiers take significantly longer times to train compared to their multilabel counterparts, showing the utility of modeling this as a multilabel problem. This is the first study that focuses on predicting RDoC constructs for biomedical literature.

keywords: Research Domain Criteria, mental illness, machine learning, document classification, Artificial Neural Networks, Support Vector Machines.

1 Introduction

Research Domain Criteria (RDoC) is an under-development framework for a more effective classification of mental illness, introduced by The National Institute of Mental Health (NIMH) [9]. Current clinical approaches to mental illness classification such as ICD-10 [12] and DSM-V [2] are primarily dependent on the signs and symptoms, which tend to overlook the underlying mechanisms of brain disorders [5]. Therefore, they fail to yield results similar to those found in recent developments in genetics and neuroscience. The RDoC approach employs more comprehensive measures taking into account neuroscience of the brain, molecular biology, and behavioral science, among many others, to analyze mental disorders [8].

Figure 1: A part of the Positive Valence Systems domain. The term specificity increases downward. Level 1 and Level 2 constructs are depicted in blue and green, respectively.

The National Institute of Mental Health has developed a matrix for aggregating RDoC data[1]. The rows of the RDoC matrix represent different constructs/categories of mental illness, while the columns represent different methods or units of analysis (such as molecules and cells) used to measure the extent to which a patient can be diagnosed with certain RDoC categories. These constructs (39 in total) are grouped into five different domains of interest, where each domain contains a number of constructs that are closely related. For example, *Reward Learning* construct in the *Positive Valence Systems* domain refers to "a process by which organisms acquire information about stimuli,

[1] https://www.nimh.nih.gov/research-priorities/rdoc/

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

actions, and contexts that predict positive outcomes, and by which behavior is modified when a novel reward occurs or outcomes are better than expected.". Part of the Positive Valence Systems domain is depicted in Figure 1. The content of this matrix is manually updated periodically by domain experts.

In order to facilitate mental illness research and advance the expansion and/or refinement of the RDoC framework, all existing biomedical documents need to be curated with RDoC concepts. Given how expensive manual curation of articles is, the ability to automatically curate biomedical articles with RDoC concepts will be crucial [6]. Therefore, in this study, we tackle this problem with natural language processing (NLP) and Machine Learning (ML) techniques to conduct document classification experiments in order to examine the feasibility of automating this task. We model this task as a supervised learning problem in which biomedical article abstracts are used as examples and the class labels are the RDoC constructs. We apply several popular supervised learning algorithms to this data and demonstrate their high-level performance in both binary and multi-label classification settings. To the best of our knowledge, this is the first study on automated prediction of RDoC constructs for biomedical literature. The outcomes of this study have implications for the various groups including psychiatrists as well as other practitioners who are interested in automated tools for RDoC.

Although there are no previous attempts at automated biomedical text classification with RDoC data, there has been many prior studies on performing document classification with similar types of data and ontologies [4, 11, 7, 15, 14]. Some of these studies include biomedical text classification on data from the TREC 2005 Genomics track with an SVM classifier using bag-of-words and biological entity names as features [4]. Also, text classification based on journal names for a dataset of biomedical articles was performed by Mishra et al. [11], using an SVM classifier with features based on concept graphs, which have the advantage of containing the semantic relationships between the features. In other similar tasks, authors used Naive Bayes classifiers with a simple bag-of-words representation, in addition to sentiment analysis features and chi-squared feature selection [7] to determine whether a thread in an online health forum needs moderators' assistance. Similarly, Wang et al. [15] uses Naive Bayes to determine the relevancy level of articles to immune epitopes. Most interestingly, tagging biomedical articles with Medical Subject Headings (MeSH) terms was attempted with deep learning elsewhere [14], where a convolutional neural network (CNN) was used to achieve a significant improvement over other traditional approaches.

The rest of the paper is organized as follows: Section 2 describes the data, features and models used as well as the experimental setup. Section 3 discusses the key observations from the experiments, and Section 4 presents conclusions and future directions.

2 Methodology

2.1 Data

As mentioned above, we formulate the task of automated text classification with RDoC as a supervised learning problem. We obtained a labeled data set of 42,936 Medline abstracts manually curated by human curators at National Alliance on Mental illness (NAMI) Montana. Each one of these has been manually labeled with at least one of 26 RDoC concepts used for this study; about 5% of the abstracts are annotated to more than one construct. A list of RDoC constructs is indexed in Table 1, showing the number of articles for each construct and to which domain each construct belongs. Some of the figures in later sections will refer to the constructs by their indices in Table 1.

2.2 Preprocessing and Features

We apply three basic steps to preprocess our dataset: 1) Stemming, 2) Removing Stop words, and 3) Removing non-ASCII characters. These steps help reduce the dimensionality of the feature space. Stemming combines different variants of each word into one standard form. Stop words is a small set of very common terms that is removed due to their limited information nature.

After performing the preprocessing steps, we transform our datasets using a bag-of-words model, where each feature indicates the presence of a word in a predefined vocabulary. In our case, the vocabulary is all the unique words in our dataset.

2.3 Models

Given that 5% of the abstracts in our dataset are annotated to more than one label, we attempt to solve this problem using both (a) binary classification and (b) multilabel classification approaches. In the first stage, we apply five supervised learning algorithms: 1) Artificial Neural Networks (ANNs), 2) Support Vector Machines (SVMs), 3) Logistic Regression (LR), 4) Decision Trees (DTs), and 5) Naive Bayes (NB). In this initial set of experiments, we use the aforementioned algorithms to perform the binary and multilabel tasks and assess their effectiveness to consider them for

Domain	Index	Construct	Level	# Articles
Negative Valence Systems	1	Potential Threat	1	1,919
	2	Sustained Threat	1	1,949
	3	Loss	1	1,900
	4	Frustrative Nonreward	1	2,153
Positive Valence Systems	5	Approach Motivation: Effort Valuation/Willingness to Work	2	241
	6	Approach Motivation: Expectancy Reward/Prediction Error	2	595
	7	Approach Motivation: Action Selection/Preference Based Decision Making	2	302
	8	Initial Responsiveness to Reward Attainment	1	513
	9	Sustained Longer Term Responsiveness to Reward Attainment	1	1,891
	10	Reward Learning	1	1,904
	11	Habit	1	2,055
Cognitive Systems	12	Attention	1	2,017
	13	Perception	1	2,996
	14	Declarative Memory	1	2,134
	15	Language	1	2,001
	16	Cognitive Control	1	2,266
	17	Working Memory	1	2,011
Social Processes	18	Affiliation and Attachment	1	2,436
	19	Social Communication	1	1,962
	20	Perception and Understanding of Self: Agency	2	1,018
	21	Perception and Understanding of Self: Self Knowledge	2	2,049
	22	Perceptions and Understanding of Others: Animacy Perception	2	309
	23	Perceptions and Understanding of Others: Action Perception	2	2,103
	24	Perceptions and Understanding of Others: Understanding Mental States	2	560
Arousal and Regulatory Systems	25	Circadian Rhythms	1	2,966
	26	Sleep Wakefulness	1	3,581

Table 1: Summary of RDoC data used in this study. The constructs are grouped by domain to show related concepts. Level 2 constructs are more specific, while level 1 construct are more general.

further analysis.

2.4 Model and Feature Selection

The second stage of this process is to take the best multilabel classifiers, selected after conducting the paired t-tests, and optimize their performance using feature and model selection. Outcomes of this stage demonstrates a more genralizable accuracy of our models.

We perform a comprehensive grid search, based on feature transformers, as well as learning algorithms parameters. A subset of these parameters that will likely influence the performance is chosen. These parameters include:

- Features: {Bag-of-Words n-gram range:((1,1), (1,2), (1,3))}
- ANN : {Activation function:(relu, tanh, logistic), Network Architecture: ((5,2),(10,3))}

The n-gram range specifies the minimum and maximum of how many words represent a single feature. For example, (1,2) indicates that the features will be single and two word features. Obviously, this option can increase the feature space significantly. Therefore, we limit the feature size to the 125,000 most occurring features.

The parameters of the learning algorithm that were selected are three different activation functions and two different network architectures for ANNs. The first number in network architecture options represents the number of hidden units in each hidden layer, while the second number gives the count of the hidden layers in those networks. We perform a grid search with this set of parameters in a five-fold nested cross validation setting to report on the optimized performance of the tuned classifiers.

2.5 Experimental Setup and Evaluation

We evaluate the performance of each classifier using their AUROC (Area Under the Receiver Operating Characteristic Curve) scores [3] averaged over a 5-fold cross validation setting [1]. Ideal performance corresponds to a score of 1, while the performance of a random classifier corresponds to a score of 0.5. In order to compare the overall performance of the classifiers, we use the Macro AUROC score, which is defined as the AUROC score averaged across the RDoC constructs.

In addition to this, we apply a paired t-test between each pair of the 5 multilabel classifiers, and report the p-values of each comparison.

We use linear kernels with SVMs in this study

because in a preliminary experiment we observed that Gaussian kernels take at least five-fold the training time of linear kernels while producing very similar performance to linear kernels (data not shown). All the learning algorithms were trained and evaluated using scikit-learn toolkit [13]. All the experiments were executed on a machine running Fedora Linux operating system with Inter Xeon 3.7 GHz processor.

3 Results and Discussion

We note that except for the ANNs, the binary classifiers considerably outperform their multilabel counterparts (Figure 2). Although these results suggest using binary classifiers is better suited for this problem, they will be very limited in how much they can improve. On the other hand, multilabel classifiers have the capability to learn the inter-relationships between different labels, which puts them at an advantage with regards to optimizing their performance with model tuning. Also, using a stack of binary classifiers will not be nearly as time-efficient as the multilabel ones (Figure 3). And when it comes to multilabel classifiers, ANN is the clear winner, which significantly outperforms all other classifiers (Table 2). Based on the above observations, we used ANN multilabel classifiers for the rest of our experiments and analysis.

Another key observation is that, the less frequent a label (i.e. construct) is, the easier for the ANN classifiers to make predictions for it (Figure 4). We can see that the least frequent constructs (i.e. in the range 0-1000) in Figure 4 correspond directly to those that are more specific (i.e. level 2) in Figure 5. Thus, the specificity of terms can explain the seemingly counter-intuitive results presented in Figure 4, noting that abstracts labeled with more specific constructs likely have more specific information in them which makes it relatively easier to learn than the more general (i.e. level 1) constructs.

Category	SVM	LR	DT	NB
ANN	8.96E-05	1.12E-04	1.00E-09	2.29E-09
SVM	-	4.52E-01	3.96E-08	3.77E-04
LR	-	-	3.49E-08	1.92E-04
DT	-	-	-	7.20E-05

Table 2: P-values obtained through two-tail paired t-tests for all the multilabel classifiers pairs. ANN: Artificial Neural Networks, SVM: Support Vector Machines, LR: Logistic Regression, DT: Decision Trees, and NB: Naive Bayes. We use 0.05 as our alpha/significance level.

As mentioned earlier, in order to obtain more generalizable and robust performance, we carried out a nested

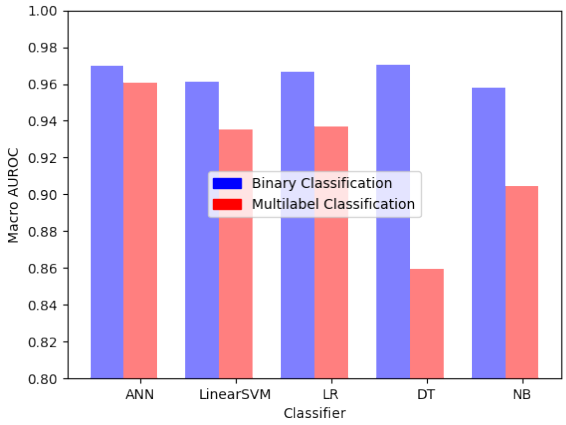


Figure 2: Macro averaged Area Under Receiver Operating Characteristic curve (AUROC) scores for binary and multilabel classifiers. ANN: Artificial Neural Networks, SVM: Support Vector Machines, LR: Logistic Regression, DT: Decision Trees, and NB: Naive Bayes.

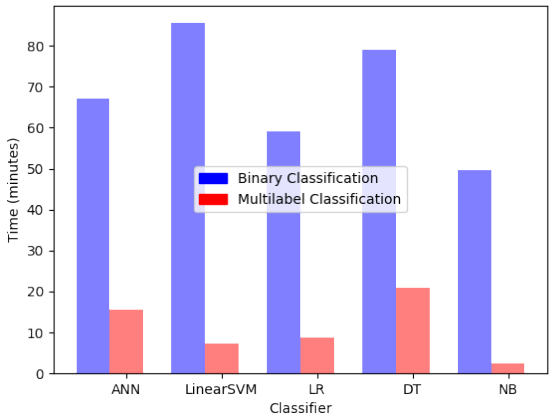


Figure 3: Runtime for each binary and multilabel classifier. ANN: Artificial Neural Networks, SVM: Support Vector Machines, LR: Logistic Regression, DT: Decision Trees, and NB: Naive Bayes.

cross validation procedure with model selection using multilabel ANNs (Figure 6). The individual AUROCs for all constructs surpass 90% AUROC. The following is the most frequently used parameter combination during this process: n-gram range - (1,2), activation function - relu and architecture - (10,3).

Similarly, an analogous experiment was performed with separate ANN classifiers trained for each of the five domains (i.e. domain specific classifiers), to determine if performance for some domains can be further im-

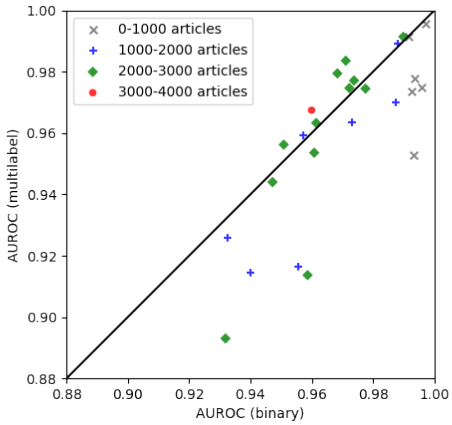


Figure 4: Scatter plot showing Area Under Receiver Operating Characteristic curve (AUROC) scores for different ranges of article counts for the binary and multilabel Artificial Neural Networks (ANN) classifiers.

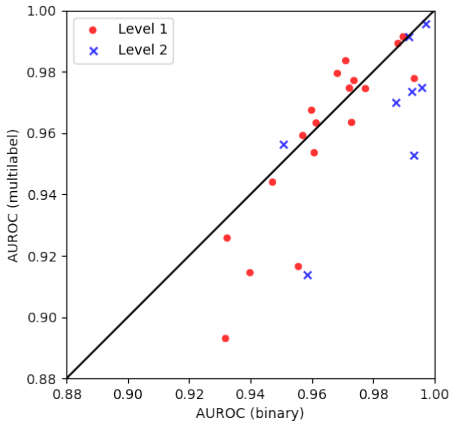


Figure 5: Scatter plot showing Area Under Receiver Operating Characteristic curve (AUROC) scores by level of constructs for the binary and multilabel Artificial Neural Networks (ANN) classifiers.

proved with such approach (Figure 7). Although, using domain specific classifiers for this task did not present an overall improvement, it did provide improvements in some of the individual constructs, most of which lie under the cognitive systems domain (constructs 12-17). However, performance of few of the other constructs declined (e.g. Reward Learning).

4 Conclusions and Future Work

In this work, we perform the first study on document classification with RDoC constructs. Thorough a series of experiments we demonstrate that overall, applying text classification with RDoC concepts in biomedical articles is very viable alternative to manual curation. Although this work employs standard methods to the problem, the excellent results indicate that automating this process can be accomplished, which can aid researchers interested in studying mental disorders from the RDoC vantage.

One of the interesting results was that the more specific RDoC concepts (i.e. level 2) were easier to predict, even though they appear significantly less frequently. We expect that the same difficulty in identifying the general constructs (i.e level 1) affected the manual curation of these articles.

There is still a considerable room for improvement with regards to this problem. First, we plan to improve our models by introducing task-specific engineered features. As reported elsewhere [15], using less strict stemming, and adding MeSH [10] terms could improve the performance of the classifiers. Other types of features that have the potential to improve the RDoC classification problem include using a set of elements from the RDoC matrix, molecules for instance, as features, which is similar to what was done elsewhere [4]. In addition, we will consider formulating this task as a structured prediction problem, given the hierarchical structure of the RDoC framework.

Furthermore, we aim to expand this study and explore the feasibility of developing a complete biocuration pipeline for RDoC. Given the high performance of the ANNs, we plan to incorporate neural networks with deep architectures and word/sentence/paragraph embeddings which would likely further improve the overall performance. It would also be very interesting to apply a topic modeling technique to identify a list of words/topics that can be used as features to further improve the performance.

5 Acknowledgements

The authors would like to thank Matt Kuntz from the National Alliance on Mental Illness (NAMI) Montana for providing the manually curated gold standard dataset of labeled Medline abstracts.

References

[1] Sylvain Arlot and Alain Celisse. A survey of cross-validation procedures for model selection. *arXiv preprint arXiv:0907.4728*, 2009.

[2] American Psychiatric Association et al. *Diagnostic and statistical manual of mental disorders (DSM-5®)*. American Psychiatric Pub, 2013.

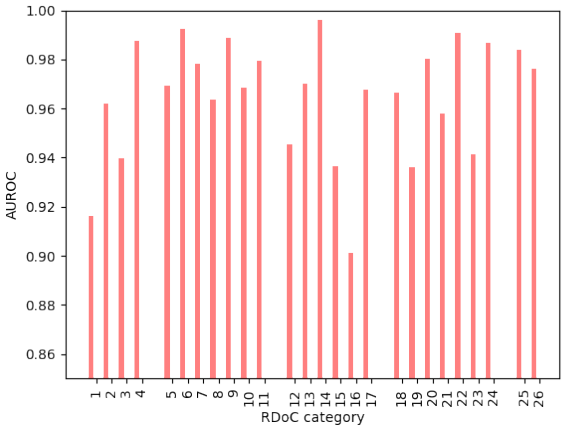


Figure 6: Optimized performance for multilabel Artificial Neural Networks (ANN) classifier showing the Area Under Receiver Operating Characteristic curve (AUROC) for each of the 26 RDoC constructs.

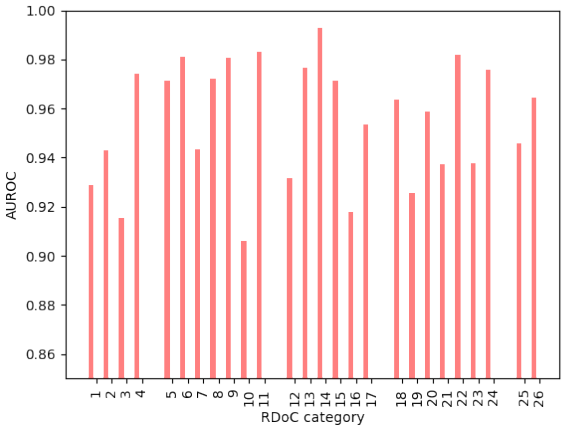


Figure 7: Optimized performance with separate multilabel Artificial Neural Networks (ANN) classifiers for each domain showing the Area Under Receiver Operating Characteristic curve (AUROC) for each of the 26 RDoC constructs.

[3] Viv Bewick, Liz Cheek, and Jonathan Ball. Statistics review 13: Receiver operating characteristic curves. *Critical Care*, 8(6):508, Nov 2004.

[4] Aaron M Cohen. An effective general purpose approach for automated biomedical document classification. *AMIA Annual Symposium proceedings. AMIA Symposium*, 2006:161–5, 2006.

[5] Bruce N Cuthbert. The RDoC framework: facilitating transition from ICD/DSM to dimensional approaches that integrate neuroscience and psychopathology. *World Psychiatry*, 13(1):28–35, 2014.

[6] Lynette Hirschman et al. Text mining for the biocuration workflow. *Database*, 2012:bas020, 2012.

[7] Jina Huh et al. Text classification for assisting moderators in online health communities. *Journal of Biomedical Informatics*, 46(6):998–1005, dec 2013.

[8] Thomas Insel et al. Research Domain Criteria (RDoC): Toward a New Classification Framework for Research on Mental Disorders. *American Journal of Psychiatry*, 167(7):748–751, jul 2010.

[9] Thomas R Insel. The NIMH research domain criteria RDoC project: precision medicine for psychiatry. *American Journal of Psychiatry*, 171(4):395–397, 2014.

[10] Carolyn E Lipscomb. Medical subject headings MeSH. *Bulletin of the Medical Library Association*, 88(3):265, 2000.

[11] Meenakshi Mishra, Jun Huan, Said Bleik, et al. Biomedical text categorization with concept graph representations using a controlled vocabulary. In *Proceedings of the 11th International Workshop on Data Mining in Bioinformatics - BIOKDD '12*, pages 26–32, New York, New York, USA, 2012. ACM Press.

[12] World Health Organization. *The ICD-10 classification of mental and behavioural disorders: clinical descriptions and diagnostic guidelines*, volume 1. World Health Organization, 1992.

[13] F. Pedregosa et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[14] Anthony Rios and Ramakanth Kavuluru. Convolutional neural networks for biomedical text classification. In *Proceedings of the 6th ACM Conference on Bioinformatics, Computational Biology and Health Informatics - BCB '15*, pages 258–267, New York, New York, USA, 2015. ACM Press.

[15] Peng Wang, Alexander A Morgan, Qing Zhang, Alessandro Sette, and Bjoern Peters. Automating document classification for the Immune Epitope Database. *BMC Bioinformatics*, 8(1):269, jul 2007.

The Flashing-Decision-Trees: Towards an Intelligent Seizure Prediction System

Arwa Ali Al-Rubaian
Computer Science Department, College of Computer and Information Science, King Saud University
Riyadh, Saudi Arabia
aalrubaian@ksu.edu.sa

Ghada Badr
IRI -The City of Scientific Research and Technological Science Applications
Alex, Egypt
badrghada@hotmail.com

Abstract

Epileptic seizures are abnormal discharges of neuronal populations causing sudden disturbances in the brain's cerebral electrical activities. The electrical activity of the brain is measured via sensors placed on the scalp, or implanted on the brain tissue that produce a stream of non-stationary Electroencephalography (EEG). Epileptic seizures are well known to have high inter and intra-patient variability, making their classification via EEG a complex task.

The aim of this research is to design and develop a real-time seizure prediction model (*Flashing Decision Trees)* that is capable of identifying seizures before their onset. The *Flashing-Decision-Trees classification model for EEG data (FDT-EEG)* diminishes the classification errors that are caused by the intra-patient variability of the seizure patterns by exploiting and adapting two bio-inspired metaheuristics, namely the *genetic algorithm* and the *firefly optimization* at different stages of the proposed model.

Keywords: Seizure detection, Epilepsy, Electroencephalography (EEG), Bio-inspired metaheuristics, firefly optimization, genetic algorithm.

1 Introduction

Epilepsy is one of the prevalent and diverse neurological disorders affecting people of all ages. Approximately 50 million people worldwide are diagnosed with epilepsy, in which 30% of these patients don't respond to medication [1]. Epilepsy is a chronic disorder of the brain characterized by recurrent unprovoked seizures. A seizure can be defined as a brief episode of uncontrolled excessive activity of a part or all of the central nervous system that may cause abnormal movements or behavior sometimes accompanied by loss of conscious [2]. These unpredictable seizures are a major source of anxiety for individuals with epilepsy due to the possibility of experiencing injuries or even life-threating situations if an episode strikes in the middle of a critical daily life activity such as driving or swimming. As a result, many researchers devoted their efforts toward evaluating seizures and predicting its occurrences, with the motivation that any system capable of predicting the occurrence of seizures in advance can improve the therapeutic treatments and improve the quality of epilepsy patients' life by helping them to adjust their preventive behavior.

Electroencephalography (EEG) is the recording of electrical activity along the scalp. It is the most convenient technique for predicting epileptic seizure episodes [3].

In this research, we propose a novel self-learnable seizure prediction system capable of discovering on-line patient specific classification models. The learning phase of the model uses two bio-inspired metaheuristics, namely the genetic algorithm [4] and the firefly optimization algorithm [5] that both have been adapted to fit the classification task.

The remaining of this paper is organized as follows: The following section describes the methodology and the basic algorithm of the proposed model. Section three summarizes the experimental results. The last section concludes our work.

2 Methodology

In this section we explain the Flashing-Decision Trees algorithm, which is an adaptable self-learnable classification model that has the required intelligence to produce patient specific classifiers capable of predicting epileptic seizures before their occurrence. The proposed approach solves both the problem of intra-patient variability and the class imbalance problem resulting in a more accurate classifier. Moreover, the pre-processing, feature extraction, and classification process uses techniques having minimal time complexity, allowing the classifier to provide results in real-time.

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

2.1 Data Acquisition

The Intracranial EEG (iEEG) data used in this study was recorded from five epileptic canines with naturally occurring generalized seizures. The dogs were maintained on anti-epileptic medications during this study. The recorded signals were read and transmitted by an implanted mobile intracranial EEG monitoring devices.

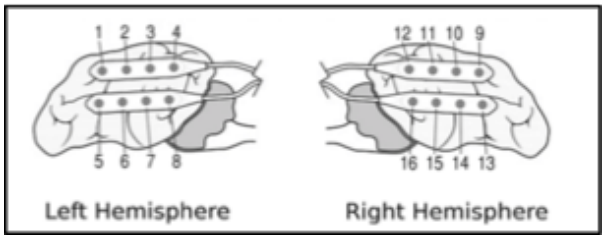


Figure1. Approximate placement of the 16 implanted electrodes [6]

The implanted telemetry device contains a bilateral of 16 electrodes arranged in a set of two strips of 4 electrodes that were placed on either side of the cortex. All iEEG data was recorded at 400Hz. The electrodes placement is illustrated in Figure1.

The data is sequentially recorded and organized into 600 -second data segments, each segment belonging to a single seizure stage. Expert readers annotated the entire iEEG data. Provided preictal segments cover one hour preceding the seizure onset, leaving out a 5 minute horizon directly before the seizure to ensure that seizures can be predicted with enough time for the subject to take appropriate action.

This dataset was developed by the University of Pennsylvania and the Mayo Clinic and is freely available on the iEEG portal.[6]

Several clinical studies have highlighted the high similarities between the symptoms and underlying nature of elliptical episodes experienced by dogs and humans[7-9]. Due to the limited and restricted availability of annotated human EEG signals, we used the elliptical canines for the proof of concept.

2.2 Feature Extraction

First, the iEEG stream was segmented using a 200-second sliding window. Each window was preprocessed in three different ways:

Differential Window:

To remove artifacts, a low pass Gaussian filter with a frequency equal to the EEG sampling frequency was applied separately on every channel of the raw EEG signals. Then, the second derivate of each channel was calculated separately using a normalization factor of 1/1000.

Fourier Transformation:

Fourier transformation was applied on each channel separately. It is clearly evident that the similarity between all EEG signals tends to increase as they reach their Direct Current (DC) component, making the higher frequencies more descriptive. Thus to decrease the dimensionality only d largest frequencies where used to extract the feature vector (where d is a design parameter that have been set to provide the maximum efficiency).

Wavelet Transformation:

Wavelets are mini waves that tend to fade to zero. The concept was first introduces by Haar in the early 1900s [10]. Adeli et. [11] Observed that the Daubechies order 4-wavelet is the most suitable mother wavelet for EEG analysis. In our research, continuous wavelet transformation was applied separately on every channel, using the Daubechies order 4-wavelet as mother wavelet.

Second, three feature vectors (one for each preprocessing method) was created. Each feature vector consists of appropriate statistical measures measured over each sliding window.

A genetic-based feature selection approach adopted from the work proposed by Oluleye B. etc. [4] is then used to choose the most significant features for the current patient from each class of features.

2.3 Classification Model

The C4.5 algorithm [12] is used to construct three different rule-based classification models, each using a single subset of features obtained from the genetic algorithm. The single decision trees are formulated as a decision forest, which is then combined and reduced to a single classification tree using a firefly optimization algorithm that is adapted to suit the classification task.

2.4 The firefly optimization phase

The firefly optimization algorithm (FA) [5] is a population-based algorithm that mimics the flashing and communication behavior of fireflies. In the simulating algorithm, every firefly represents a full decision tree and a potential classification model. The amount of light produced by the firefly is proportional to its fitness function.

$$Fitness\ (FF_i) = accuracyRatio\ x\ (\#\ true\ positive + \#\ true\ negative)/\ total\ number\ of\ instances + \#true\ positive/\ number\ of\ preictal\ x\ sensitivityRatio \quad (1)$$

The fitness function in our proposed model is given by equation (1) in which a higher ratio is assigned to the accuracy. The accuracy and sensitivity ratios are randomly selected at each iteration with restricting the sensitivity ration to be in the range [0, 0.4] and the accuracy ratio to be the complement of the sensitivity ratio calculated as: 1 – sensitivity ratio.

Fireflies are attracted, and thus move towards brighter fireflies. The amount of light a firefly perceives is inversely proportional to the distance between the two fireflies, due to natural phenomena of light loss as it travels through a horizon. The base algorithm is listed in Algorithm1 .[5]

Algorithm 1: Firefly optimization

```
generate initial population of
fireflies xi(i=1,2,3,...,n)
while (t < MaxIterations)
    Measure and update the light
    intensity of all  fireflies.
    Sort the fireflies (descending
    order) based on their light
    intensity
    for i=1:n (all fireflies)
         for j=1:n (all fireflies)
           if (LightIntensityi <
         LightIntensityj –
         Distance(fireflyi, fireflyj)
         && fireflyi has moved <
         maxAllowedMovment)
            Move fireflyi towards
fireflyj
            End If
         End for j
     End for i
End while
Rank all fireflies and find the best
solution
```

Finally, upon termination of the firefly algorithm the firefly having the highest fitness is announced as the resulting combined decision tree and is then used to classify new data instances into interictal or preictal signals.

3 Experimental Setup

Experiments were conducted on all five subjects of the dataset explained in section 2.1. Two thirds of each subject's EEG stream is used for training the model and the remaining third is reserved as a blind testing sample. The performance measures are based on ten different executions. All classifiers used in the comparison are given the same feature set (chosen by the genetic algorithm) as an input.

4 Experimental Results

Compared to other work in the literature the *Flashing-Decision Trees algorithm for EEG signals (FDT-EEG)* have proven its effectiveness in balancing the sensitivity and specificity ratios regardless of how great the imbalance between the classes was in the original data. Although the work proposed by An.J etc. [10] selects a poor sample of the data consisting of 5 hours of preictal recordings along with equal duration of interictal data (i.e balancing the classes prior to the learning phase), and the *FDT-EEG* uses the entire recordings consisting of 5 hours of preictal signals and 75 hours of interictal data, the results are almost comparable. The FDT-EEG model was able to tackle the class imbalance problem; providing even better overall accuracy, and falling back by around only 10% in sensitivity.

Moreover, various experiments conducted on the same data set have proven that given a feature set, the *FDT-EEG* model constructs a more balanced and effective model than its benchmark competitors: The Weka implementation of the C4.5 decision tree based classifier allowing pruning with a confidence factor of 0.25 [12] and the Weka implementation of the cost sensitive support vector machine classifier LIB-SVM with a Radial Basis Function kernel of degree 3 [14]. Performance measures including accuracy, sensitivity, and specificity detailed in Table 1 have been used to compare these models.

Table 1 Performance comparison of the FDT-EEG with benchmark classifiers

Subject	Classifier	Accuracy	Sensitivity	Specificity
Dog 1	**C4.5**	88.89%	0.00%	93.33%
	SVM	95.24%	0.00%	100%
	FDT-EEG (Best)	88.49 %	20.83 %	91.88 %
	FDT-EEG (Average)	87.66 %	10.83 %	91.50 %
Dog 2	**C4.5**	92.82%	21.43%	98.80%
	SVM	92.27%	0.00%	100%
	FDT-EEG (Best)	82.32 %	33.33 %	86.43 %

	FDT-EEG (Average)	83.02 %	16.90 %	88.56 %
Dog 3	**C4.5**	93.78%	0.00%	98.47%
	SVM	95.24%	0.00%	100%
	FDT-EEG (Best)	92.46 %	16.67 %	96.25 %
	FDT-EEG (Average)	91.49 %	9.03 %	95.62 %
Dog 4	**C4.5**	85.60%	8.08%	95.15%
	SVM	89.04%	0.00%	100%
	FDT-EEG (Best)	86.27 %	21.21 %	94.28 %
	FDT-EEG (Average)	84.46 %	13.23 %	93.23 %
Dog 5	**C4.5**	91.46%	26.67%	95.78%
	SVM	93.75%	0.00%	100%
	FDT-EEG (Best)	92.08 %	66.67 %	93.78 %
	FDT-EEG (Average)	91.88 %	41.67 %	95.22 %

The final classification accuracy of the produced model is highly dependent on the used feature vector. In which the FDT-EEG heuristically selects an optimal highly descriptive, patient specific feature vector using the genetic algorithm. Moreover, the simple IF- THEN representation of the final model adds understandability to it, and allows the justifiability of the classification results.

5 Conclusion

In this research we proposed a novel seizure prediction system *(FDT-EEG)* that combines metaheuristics namely, the genetic algorithm and the firefly optimization algorithm along with the rule-based classifier to produce an intelligent classification model capable of transferring and customizing a base classification model into patient specific classifiers, diminishing the classification error caused by the broad intra-patient variability of the seizure patterns. Several modifications were applied on both metaheuristics to fit their purpose in the proposed model. Results and observations prove the effectiveness of the proposed model in the classification of class imbalanced problem especially when the minor class is of much more importance, as in the problem of seizure prediction.

As for future work, The Flashing-Decision-Trees classification can be enhanced in several ways. We would like to investigate the effect of changing the formulation of the initial population of the firefly optimization step making it consist of initial decision trees constructed from a single rule. We would also like to investigate the possibility of using other linear bimodal feature selection methods and bivariate features such as pearson correlation that may provide better description of the preictal signals.

Obtaining real clinical data from epileptic patients and using it to evaluate the performance of the *FDT-EEG* model, and measure its applicability in remote health care is of high interest, and is among the near future plans.

References

1. W. H. Organization, “WHO | Epilepsy,” 2015.
2. “Epilepsy Information Page: National Institute of Neurological Disorders and Stroke (NINDS).” [Online]. Available: http://www.ninds.nih.gov/disorders/epilepsy/epilepsy.htm. [Accessed: 19-Oct- 2015].
3. “Types of Seizures | The Johns Hopkins Epilepsy Center.” [Online]. Available: http://www.hopkinsmedicine.org/neurology_neurosurgery/centers_clinics/epilepsy/seizures/types/. [Accessed: 19-Oct-2015].
4. B. Oluleye, A. Leisa, and D. Dean, “A Genetic Algorithm-Based Feature Selection,” Br. J. Math. Comput. Sci., vol. 5, no. 4, pp. 899–905, 2014.
5. X. Yang, “Firefly Algorithms for Multimodal Optimization,” pp. 169–178, 2009.
6. “iEEG portal.”
7. Berendt, M., Gulløv, C. H., Fredholm, M. (2009) Focal epilepsy in the Belgian shepherd: evidence for simple Mendelian inheritance. Journal of Small Animal Practice 50, 655-661
8. Weissl, J., Hülsmeyer, V., Brauer, C., et al. (2012) Disease progression and treatment response of idiopathic epilepsy in Australian Shepherd dogs. Journal of Veterinary Internal Medicine 26, 116-125
9. Uriarte A, Maestro Saiz I. Canine versus human epilepsy: are we up to date? J Small Anim Pract 2016;57:115–121.
10. C. Charles K., An Introduction to Wavelets. Academic Press, San Diego, 1992.
11. H. Adeli, Z. Zhou, and N. Dadmehr, “Analysis of EEG records in an epileptic patient using wavelet transform,” vol. 123, pp. 69–87, 2003.
12. H. Du, Data Mining Techniques and Applications An Introduction. Cengage Learning, 2010.
13. J. An, A. Bearman, and C. Dong, “Predicting Seizure Onset in Epileptic Patients Using Intracranial EEG Recordings.”
14. C. Chang, C. Lin, LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology, vol. 2, 2011.

K-means-based Feature Learning for Protein Sequence Classification

Paul Melman and Usman W. Roshan
Department of Computer Science, NJIT
Newark, NJ, 07102, USA
pm462@njit.edu, usman.w.roshan@njit.edu

Abstract

Protein sequence classification has been a major challenge in bioinformatics and related fields for some time and remains so today. Due to the complexity and volume of protein data, algorithmic techniques such as sequence alignment are often unsuitable due to time and memory constraints. Heuristic methods based on machine learning are the dominant technique for classifying large sets of protein data. In recent years, unsupervised deep learning techniques have garnered significant attention in various domains of classification tasks, but especially for image data. In this study, we adapt a k-means-based deep learning approach that was originally developed for image classification to classify protein sequence data. We use this unsupervised learning method to preprocess the data and create new feature vectors to be classified by a traditional supervised learning algorithm such as SVM. We find the performance of this technique to be superior to that of the spectrum kernel and empirical kernel map, and comparable to that of slower distance matrix-based approaches.

keywords: Protein classification, Unsupervised learning, K-means

1 Introduction

Identifying protein functionality is one of the principle challenges of modern biological sciences. While there do exist precise alignment techniques such as Smith-Waterman [15], due to their highly complex structures and behaviors, modeling large sets of proteins using deterministic methods is impractical, if not impossible, with currently available technology. Therefore, we must rely on heuristic techniques for analyzing proteins. One way to elucidate the behavior of a protein is to compare it to proteins with known properties, via protein classification [14].

Over the course of evolutionary history, genes and proteins with similar functionality diverge due to the accumulation of mutations. As a result, analogous proteins may become difficult to recognize. Through the use of machine learning techniques, it is possible to find relationships between protein sequences that would otherwise be obscured.

2 Related Work

K-means is a clustering algorithm that is used to partition points into k clusters based on the nearest cluster mean, or centroid [10]. The use of K-means clustering in image classification is based upon the principle of representation learning. An image is broken up into fragments by a sliding window, and these fragments, along with fragments of all the other images in the dataset, are then clustered by similarity. The centroids of the clusters represent features learned from the images, such as corners or diagonal lines. A new feature vector representation of the original image can then be created based on the presence or absence of the various features [3] [4]. This technique has been successfully employed in image classification tasks such as distinguishing between bacterial colonies of different species or identifying weeds [6] [17].

3 Methodology

3.1 Data & Materials

The data we use in this study originate from the SCOP, CATH, COG, and 3PGK protein datasets [2] [11] [18] [13]. The datasets and classification tasks were obtained from the Protein Classification Benchmark collection [16]. There are a total of 3242 classification tasks across all the datasets. For the SVM classifier we use the Scikit-learn Python library[12]. See Table 1 for details.

The 3PGK dataset, which consists of sequences of 3-phosphoglycerate kinase from various species, has 10 tasks that consist of classifying sequences into kingdoms based on phyla.

The SCOP dataset has three standard classification task categories: Classification of sequences into super-

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

Table 1: Datasets

Dataset	# seqs	average seq len	# frags (len 14)	tasks	task types
3PGK	131	411	52137	10	1
CATH95	11373	150	1562581	1414	8
SCOP95	11944	173	1916986	1629	6
COG	17973	373	6467745	189	2

families based on families (246 tasks), classification into folds based on superfamilies (191 tasks), and classification into structural classes based on folds (377 tasks). There are also three 5-fold cross validation task sets. The first consists of 98 superfamilies with five random splits each of training and test data where the positive examples come from one superfamily and the negative examples are taken from all other superfamilies for 490 total tasks. The same 5-fold split technique was used for 58 folds for 290 total tasks, and for 7 structural classes for 35 total tasks.

The CATH dataset has four standard classification task categories: Classification into homology groups based on similarity groups, with 165 tasks; classification into topology groups based on homology groups (199 tasks), classification into architecture groups based on topology groups (297 tasks), and classification into structural classes based on architecture groups (33 tasks). There are also four 5-fold cross validation task sets: By homology (375 tasks), by topology (235 tasks), by architecture (95 tasks), and by structural class (15 tasks).

The COG dataset has two types of classification task. In the first task category, the positive training sets consist of prokaryote protein sequences representing a particular biological function (COG), and the positive test sets consist of eukaryote protein sequences representing the same COG. The negative training set consists of sequences representing other COGs. There are a total of 117 tasks of this type. The second category involves separating proteins belonging to the kingdom Archaea from proteins belonging to any other kingdom. There are 72 of these tasks.

These datasets also came with published benchmarks that were computed by creating all against all BLAST and Smith-Waterman distance matrices and an SVM classifier.

3.2 Empirical and Spectrum Kernels

The first baseline method we use for this study is the empirical kernel map. For this we use 3364 reference protein sequences from the seed pairwise alignments in PREFAB 4.0 [7]. The feature vector for each protein in the dataset is created by aligning it to each reference protein using BLAST [1]. Each dimension of the final vector is the BLAST score of the alignment to a different reference protein; therefore, the feature vector for each protein has 3364 dimensions.

The second baseline we use is the spectrum kernel, which creates a feature vector by counting the number of occurrences of every possible amino acid triplet in each sequence [9]. Both the spectrum kernel and empirical kernel methods used an SVM classifier [5].

3.3 String-based K-means Feature Learning

In this method we first produce fragments of every protein sequence in a given dataset using a sliding window approach (see Figure 1). We then cluster all fragments using a string-based k-means (see number of fragments in each dataset in Table 1). To compute the centroid of each cluster, we find the mode of each character position across all the fragments in that cluster (Figure 2).

To compute the distance from a fragment to a centroid we examine two different measures. First, we use Hamming distance, where we compare the fragment and centroid at each character position and count the number of mismatches. A larger count represent more dissimilar strings and therefore a greater distance. The second distance measure we examine is based on the BLOSUM62 matrix, which is derived from empirical observations of amino acid substitution probabilities [8]. The distance is represented by the negative of the BLOSUM62 alignment score of the two strings. The negative is used so that this algorithm optimizes for the minimization of the distances between points (fragments) and their nearest centroids, just as the traditional K-means algorithm does.

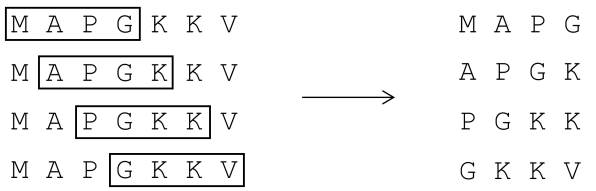


Figure 1: An example of the fragmentation process with fragment length 4 and a stride of 1.

To create a feature vector from the clusters, we use a method that is a cross between the triangle encoding and hard encoding schemes employed by Coates, Ng, and Lee [4], as described in Equation (1). For each fragment, we create a vector with k dimensions, where k is the number of clusters. For the feature f that corresponds to the index of the nearest centroid, the

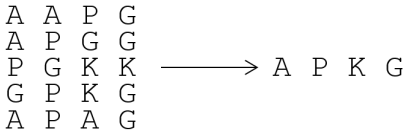


Figure 2: The centroid sequence is created by taking the mode character at each position.

Algorithm 1: String K-means Pseudocode

```
choose k random fragments as starting centroids
while i = 0; i < max_iter; i++ do
    for Each fragment do
        Find distance to each centroid
        Assign to closest centroid
    for Each cluster do
        Calculate new centroid
    if No change in centroids then
        Break
```

value is set to the mean of the distance to that centroid plus the mean of the distance to all centroids; for all other features, the value is set to zero to create a sparse vector with k dimensions. The vectors for all the fragments of a protein sequence are then sum-pooled to create the final feature vector for the sequence (Figure 3).

$$f_k(x) = \begin{cases} \mu(z) + z_k \text{ if } k = \arg\min_j \|c^{(j)} - x\|_2^2 \\ 0 \quad \text{otherwise} \end{cases} \quad (1)$$

where $z_k = \|x - c^{(k)}\|_2$.

We then train a linear support vector machine classifier on the feature vectors of the training dataset and evaluate it on the test dataset. This is done for each task in each task category based on the cast matrices obtained from the benchmark database.

4 Results

4.1 Comparison of Parameters

We ran the deep K-means algorithm with 2000, 4000, 8000, and 16,000 clusters on the CATH dataset (Figure 4) and with 2000 and 8000 clusters on COG (Figure 11). The results show a trend of improvement as the number of features increases. We also found that fragment length had little impact (Figure 5). These effects mirror those of Coates, Lee, and Ng [4]. However, unlike in Coates and Ng's later analysis of the K-means method for image classification, we did not experience

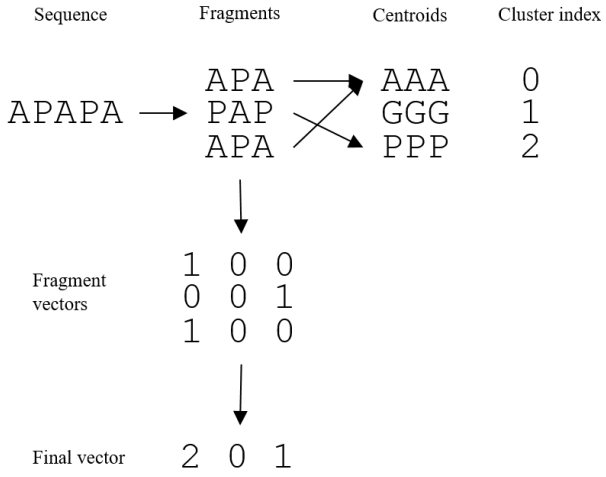


Figure 3: An illustration of how the feature vectors are created. Hamming distance and hard encoding are used in this example.

problems with imbalanced or empty clusters [3]. We found that fragments became well distributed across clusters without the need for any additional processing (Figure 13). Additionally, we found that BLOSUM distance was superior to Hamming distance (Figures 6 and 7). Clustering made up most of our run time and, as expected, run time is longer for larger datasets (Table 2).

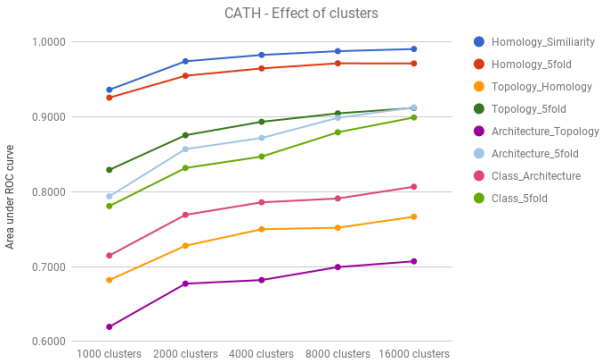


Figure 4: Effect of number of features (centroids) on CATH data.

4.2 Comparison to other Methods

We found that our K-means feature learning method outperformed the empirical and spectrum kernels on nearly every category of tasks. With 16,000 clusters, the K-means approach outperformed the empirical kernel map and the spectrum kernel on every task in

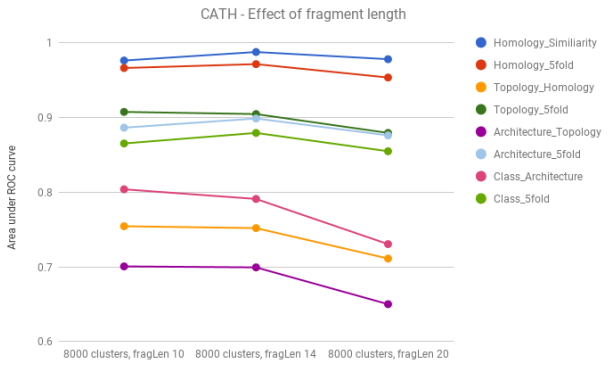


Figure 5: Effect of number of fragment length on CATH data.

Figure 6: Effect of Hamming distance vs. BLOSUM distance on 3PGK.

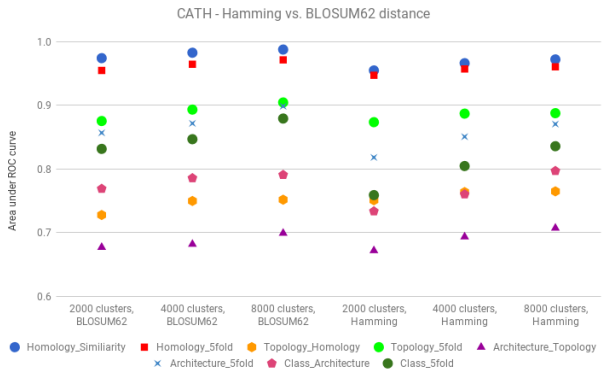


Figure 7: Effect of Hamming distance vs. BLOSUM distance on CATH.

Table 2: Runtimes for clustering on Intel Xeon E5-2630-v4 with 20 cores.

Dataset	time (minutes)
3PGK (600 clusters)	4
CATH (16k clusters)	799
SCOP (16k clusters)	1068
COG (8k clusters)	4493

Table 3: Average areas under the ROC curve for various methods. BLAST and SW are all vs. all BLAST and Smith-Waterman distance matrices classified with SVM

Dataset	K-means	Empirical	Spectrum	BLAST	SW
3PGK	0.964	0.906	0.887	0.919	0.923
CATH	0.870	0.819	0.847	0.860	0.924
SCOP	0.842	0.810	-	0.841	0.896
COG	0.949	0.910	0.944	0.923	0.931

CATH, and outperformed the all against all BLAST matrix on all the standard classification task sets, while under-performing slightly on the cross-validation sets (Figure 9). Our method also outperformed the empirical kernel on both COG sets and all but one SCOP sets (Figure 10), and beat the spectrum kernel on the Eukaryotes-Prokaryotes COG task set (Figure 11). Overall, our method had a higher average ROC AUC than all other methods on the 3PGK (Figure 8) and COG datasets, and beat all methods but all-vs-all Smith-Waterman alignment on CATH and SCOP (Table 3).

5 Discussion

Our K-means-based representation learning method performs on par with state of the art protein classification techniques. Overall, our method tends to outperform established methods on the standard protein classification tasks in the CATH and SCOP datasets, which generally seem to be more difficult by virtue of the lower performance by all methods, while slightly underperforming against BLAST similarity matrix methods on the 5-fold cross validation tasks (Figures 9 and 10). Our method under-performed against Smith-Waterman similarity matrices on CATH and SCOP, but

Table 4: Wilcoxon p-values for CATH. K-means performs significantly better than the spectrum kernel and empirical kernel map.

CATH	16k clusters	Spectrum	Empirical
16k clusters	-	0.008	<0.0001
Spectrum	0.008	-	0.003
Empirical	<0.0001	0.003	-

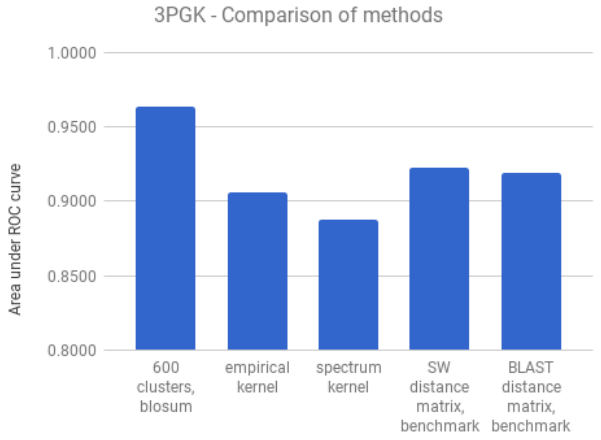


Figure 8: Comparison of K-means to empirical kernal map, spectrum kernel, and BLAST similarity matrix on 3PGK data.

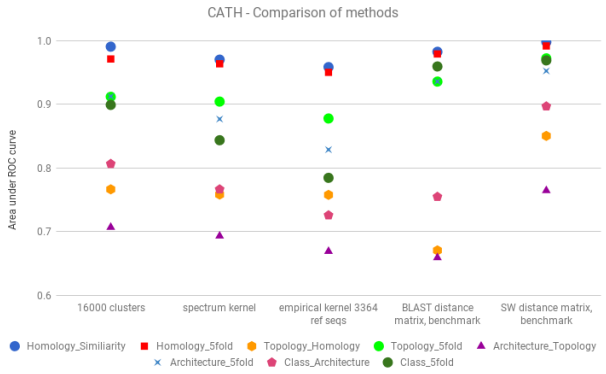


Figure 9: Comparison of K-means to empirical kernal map, spectrum kernel, and BLAST similarity matrix on CATH data.

outperformed them on COG and 3PGK. This may be due to the higher average sequence length of COG and 3PGK (as seen in Table 1).

Our method also shows promise in its ability to generalize. Despite low similarity between the CATH and COG sequence data, the features learned from the COG data were nearly as useful in learning to classify CATH proteins as features learned from CATH proteins (Figure 12). This suggests that the features being learned are generic protein features, though further testing is required to establish just how general they are. When applied to image data, K-means-based feature learning is able to learn generic visual features such as corners or lines or a particular orientation [4]. In principle, the same should be possible for proteins.

Furthermore, our method has potential utility for

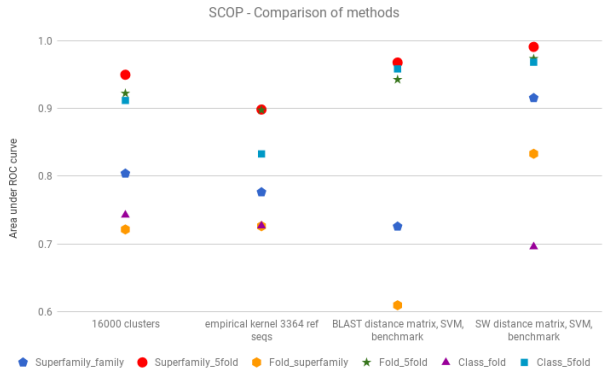


Figure 10: Comparison of K-means to empirical kernal map and BLAST and Smith-Waterman similarity matrices on SCOP data.

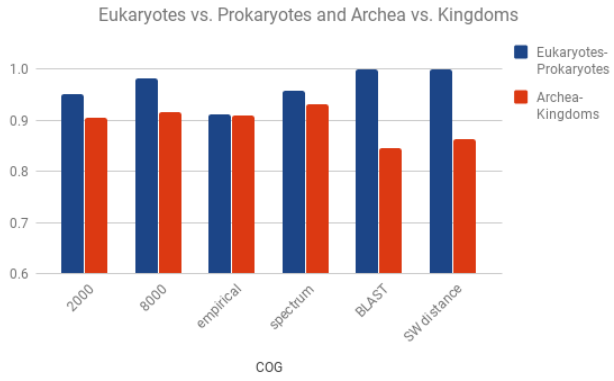


Figure 11: Results on COG data.

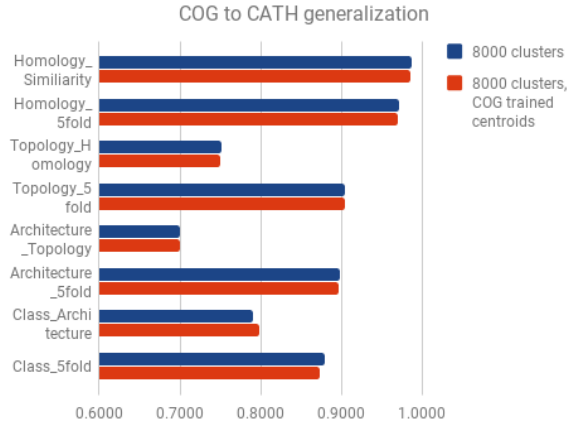


Figure 12: Comparison of classification performance on CATH dataset of centroids trained on CATH vs. centroids trained on COG.

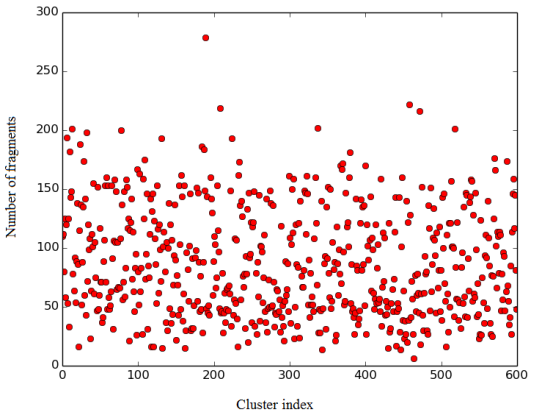


Figure 13: Distribution of 3PGK fragments across 600 clusters.

protein alignment as well. It may be possible to use it to classify and rank alignments to find the best ones.

Our code for fragmenting, clustering, and classifying protein sequences is available at `https://web.njit.edu/~usman/kmeans_fl_protein/`

6 Acknowledgment

We thank the NJIT Academic and Research Computing Systems Group (ARCS) for their support in running experiments for this study.

References

[1] Stephen F Altschul et al. "Basic local alignment search tool". In: *Journal of molecular biology* 215.3 (1990), pp. 403–410.

[2] Antonina Andreeva et al. "SCOP database in 2004: refinements integrate structure and sequence family data". In: *Nucleic acids research* 32.suppl_1 (2004), pp. D226–D229.

[3] Adam Coates and Andrew Y Ng. "Learning feature representations with k-means". In: (2012), pp. 561–580.

[4] Adam Coates, Andrew Ng, and Honglak Lee. "An analysis of single-layer networks in unsupervised feature learning". In: (2011), pp. 215–223.

[5] Corinna Cortes and Vladimir Vapnik. "Support-vector networks". In: *Machine learning* 20.3 (1995), pp. 273–297.

[6] Murat Dundar et al. "Simplicity of kmeans versus deepness of deep learning: A case of unsupervised feature learning with limited data". In: (2015), pp. 883–888.

[7] Robert C Edgar. "MUSCLE: multiple sequence alignment with high accuracy and high throughput". In: *Nucleic acids research* 32.5 (2004), pp. 1792–1797.

[8] Steven Henikoff and Jorja G Henikoff. "Amino acid substitution matrices from protein blocks". In: *Proceedings of the National Academy of Sciences* 89.22 (1992), pp. 10915–10919.

[9] Christina Leslie, Eleazar Eskin, and William Stafford Noble. "The spectrum kernel: A string kernel for SVM protein classification". In: (2001), pp. 564–575.

[10] Stuart Lloyd. "Least squares quantization in PCM". In: *IEEE transactions on information theory* 28.2 (1982), pp. 129–137.

[11] Frances Pearl et al. "The CATH Domain Structure Database and related resources Gene3D and DHS provide comprehensive domain family information for genome analysis". In: *Nucleic acids research* 33.suppl_1 (2005), pp. D247–D251.

[12] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[13] J Dennis Pollack, Qianqiu Li, and Dennis K Pearl. "Taxonomic utility of a phylogenetic analysis of phosphoglycerate kinase proteins of Archaea, Bacteria, and Eukaryota: insights by Bayesian analyses". In: *Molecular phylogenetics and evolution* 35.2 (2005), pp. 420–430.

[14] Rabie Saidi, Mondher Maddouri, and Engelbert Mephu Nguifo. "Protein sequences classification by means of feature extraction with substitution matrices". In: *BMC bioinformatics* 11.1 (2010), p. 175.

[15] Temple F Smith and Michael S Waterman. "Identification of common molecular subsequences". In: *Journal of molecular biology* 147.1 (1981), pp. 195–197.

[16] Paolo Sonego et al. "A protein classification benchmark collection for machine learning". In: *Nucleic Acids Research* 35.suppl_1 (2006), pp. D232–D236.

[17] JingLei Tang et al. "Weed identification based on K-means feature learning combined with convolutional neural network". In: *Computers and Electronics in Agriculture* 135 (2017), pp. 63–70.

[18] Roman L Tatusov et al. "The COG database: an updated version includes eukaryotes". In: *BMC bioinformatics* 4.1 (2003), p. 41.

Protein Mutation Stability Ternary Classification using Neural Networks and Rigidity Analysis

Richard Olney,[1] Aaron Tuor,[2] Filip Jagodzinski[1] and Brian Hutchinson[1,2]
[1]Western Washington University, Bellingham, WA, USA
[2]Pacific Northwest National Laboratory, Seattle, WA, USA
{filip.jagodzinski, brian.hutchinson}@wwu.edu

Abstract

Discerning how a mutation affects the stability of a protein is central to the study of a wide range of diseases. Machine learning and statistical analysis techniques can inform how to allocate limited resources to the considerable time and cost associated with wet lab mutagenesis experiments. In this work we explore the effectiveness of using a neural network classifier to predict the change in the stability of a protein due to a mutation. Assessing the accuracy of our approach is dependent on the use of experimental data about the effects of mutations performed *in vitro*. Because the experimental data is prone to discrepancies when similar experiments have been performed by multiple laboratories, the use of the data near the juncture of stabilizing and destabilizing mutations is questionable. We address this later problem via a systematic approach in which we explore the use of a three-way classification scheme with stabilizing, destabilizing, and inconclusive labels. For a systematic search of potential classification cutoff values our classifier achieved 68 percent accuracy on ternary classification for cutoff values of -0.6 and 0.7 with a low rate of classifying stabilizing as destabilizing and vice versa.

Introduction

Performing an amino acid substitution in a protein may induce a structural change that can have wide ranging effects on the protein's function. Discovering which mutations are destabilizing and which are stabilizing provides insights into many types of disorders, such as sickle cell anemia [16] and some types of cancer [7], and is important for understanding communicable and highly mutable diseases (e.g. HIV [9], influenza [17]).

In vitro experiments are necessary to determine how a mutation affects a protein's function. However, these experimental efforts come at considerable time and cost, as a single mutagenesis experiment followed by X-ray crystallography work may require weeks of wet lab work. Moreover, because each residue in a protein can in principle be one of 20 naturally occurring amino acids, the set of all possible mutations is vast, so computational tools for screening likely candidates for investigation in a wet lab setting are desired.

We explore the use of a neural network classifier for automatic inference of the effects of mutations. The ground truth, obtained from wet lab experiments recorded in the Protherm database [15], is in the form of change of the Gibbs Free Energy ($\Delta\Delta G$) indicating whether a mutation is destabilizing (negative $\Delta\Delta G$) or stabilizing (positive $\Delta\Delta G$). Typical approaches either predict the $\Delta\Delta G$ value given a specified mutation (regression) [3, 6], or predict whether a mutation is stabilizing or destabilizing (binary classification) [3].

Here we deal with ternary classification in which a third "inconclusive" class is introduced. That class is important because all available $\Delta\Delta G$ data is from wet-lab work, and as with any physical experiment, there is the chance of some inherent error. The use of a $\Delta\Delta G$ value close to 0 might cause a classifier to misclassify a stabilizing mutation as destabilizing or vice versa, if indeed the reported true label is erroneous. Mislabeled data is detrimental to training a model, so we systematically performed many computational experiments, testing the range of indeterminate values to find an optimal inconclusive range for $\Delta\Delta G$.

We trained deep neural network classification models across a systematic search of the $\Delta\Delta G$ cutoff space. Using the results of these experiments we generated confusion matrices in order to assess the utility and classification performance for each cutoff range. We found several interesting trends and potential cutoff ranges, which we present here via case studies.

Related Work and Motivation

The use of experimental stability data ($\Delta\Delta G$) is prevalent in research that aims to offer computational techniques for assessing the effects of mutations [14, 4, 12]. An often-cited source is the ProTherm

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

$\Delta\Delta G$ lower val	$\Delta\Delta G$ upper value	Num Entries
-10	10	4184
-1	1	2157
-0.5	0.5	1364
-0.1	0.1	390

Table 1: Distribution of $\Delta\Delta G$ values among ProTherm entries for which stabilizing information is available

database [15]. It provides information about the proteins, mutations performed, wet lab conditions, and stability measurements for 25,820 mutation experiments reported on in the literature. Of those ProTherm entries for which stability data is provided, the $\Delta\Delta G$ values range from about -10 kCal/mol (indicating a strongly destabilizing mutation), to approximately +10 kCal/mol (strongly stabilizing). The single inflection value of zero $\Delta\Delta G$ designates that point on the real number line where the effect of a mutation changes from stabilizing to destabilizing.

In Table 1 we show the count of entries in ProTherm for three separate ranges of $\Delta\Delta G$ values. Of the 4,184 entries with $\Delta\Delta G$ ranges between -10 and 10 kCal/mol, 1,364 of them are in the range $[-0.5, +0.5]$. Thus, a large portion of ProTherm entries are for values where experimental errors or instrument discrepancies might mean that a recorded stabilizing mutation is indeed destabilizing, and vice versa. It is for this reason that experimental data for $\Delta\Delta G$ values in the range $[-0.5, +0.5]$ is often not used.

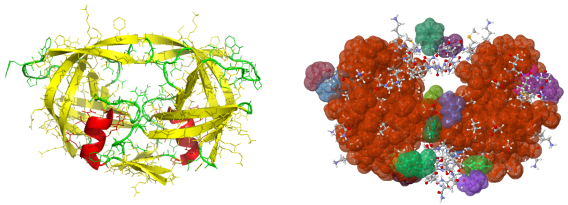
In addition, there are a number of entries in the ProTherm database where identical experiments performed by different labs have recorded opposite (stabilizing versus destabilizing) results. Two examples :

- Cold shock Protein, ProTherm Entries 21797 and 21839, $\Delta\Delta G$ = -0.05 and +0.7, respectively
- Myoglobin Sperm Whale, ProTherm Entries 2092 and 2814, $\Delta\Delta G$ = -0.9 and + 0.1, respectively

The use of $\Delta\Delta G$ data, therefore, as values for assessing and training a predictive model, must be done with care. For this reason, we report the predictive power of our machine learning model in the context of a systematic approach of varying the $\Delta\Delta G$ values designating boundaries between three classification labels.

Methodology

Here we summarize rigidity analysis and describe how we generate features and labels for training our neural network classifier machine learning model, and the experiment setup for evaluating the model.



(a) Cartoon (b) Rigidity Analysis

Figure 1: Rigidity analysis (PDB 1HVR) identifies rigid clusters. Orange is the largest cluster with 1,371 atoms.

Rigidity Analysis

To help reason about the effects of mutations, we take an approach that relies on a fast algorithm for assessing the rigidity of a protein [8, 11]. In rigidity analysis, atoms and their chemical interactions are used to construct a mechanical model, a graph is constructed from the model, and pebble game algorithms [10] are used to analyze the rigidity of the associated graph. The results are used to infer the rigid regions of the protein (Figure 1). We rely on the KINARI rigidity software for performing rigidity analysis [8].

Mutants, Rigidity Distance

To generate *in silico* mutant structures corresponding to the mutation data in ProTherm, we used our ProMuteHT [2] software. In this study, we rely on the rigidity analysis results of the wild type (non-mutated protein), and a mutant, to assess the effects of a mutation. In our previous work [1, 6], we used an $RD_{WT \rightarrow mutant}$ rigidity distance metric to quantitatively assess the impact of mutating a residue to one of the other 19 naturally occurring amino acids:

$$RD_{WT \rightarrow mutant} : \sum_{i=1}^{i=LRC} i \times [WT_i - Mut_i]$$

where WT refers to Wild Type, Mut refers to mutant, and LRC is the size of the Largest Rigid Cluster (in atoms). Each term of the summation $RD_{WT \rightarrow mutant}$ metric calculates the difference in the count of a specific cluster size, i, of the wild type and mutant, and weighs that difference by i.

Wet Lab Mutation Data – $\Delta\Delta G$

Labels ($\Delta\Delta G$) and metadata (pH, temperature, etc.) of mutations were retrieved from the ProTherm [15] database of mutation experiments. The rigidity features for each mutant and wild type were generated by rigidity analysis using the KINARI software. A total of $2,072$ data points from ProTherm meet the criteria

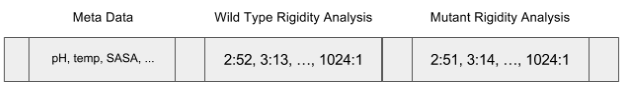


Figure 2: The form of a feature vector as input to the DNN. It consists of experiment meta data, such as Solvent Accessible Surface Area (SASA), pH, and temperature, concatenated with the rigid cluster frequencies of both wild type and mutant proteins.

for our experimental setup (i.e., single chain proteins, single mutations, any value of $\Delta\Delta G$). The input to our model is shown in Figure 2. The data set of 2,072 proteins is split into a training set of 1,438 proteins for fitting a classifier, a development set of 324 proteins for finding the best neural network configuration, and a test set of 310 proteins to test generalization error.

Deep Neural Network Classifier

A deep neural network (DNN) classifier is a parameterized function mapping a real valued vector to a probability distribution over a set of classes. We model the probability distribution over classes of mutation as stabilizing, destabilizing, or inconclusive, as a function of the rigidity analyses and experimental conditions, using a DNN with L hidden layers, $\mathbf{h}_{(1)}, \mathbf{h}_{(2)}, \ldots, \mathbf{h}_{(L)}$. This neural network classifier takes as input a feature vector $\mathbf{x}$ (Fig. 2) which we alternatively denote as as $\mathbf{h}_{(0)}$. The classifier outputs a probability vector $\mathbf{p} \in \mathbb{R}^3$, the elements of which are calculated as:

$$\mathbf{p}_k = \frac{\texttt{exp}(\mathbf{o}_k)}{\sum_{j=1}^{3} \texttt{exp}(\mathbf{o}_j)}, \text{ where} \quad (1)$$

$$\mathbf{o} = \mathbf{U}\mathbf{h}_{(L)} + \mathbf{a} \text{ and} \quad (2)$$

$$\mathbf{h}_{(\ell)} = f(\mathbf{W}_{(\ell)}\mathbf{h}_{(\ell-1)} + \mathbf{b}_{(\ell)}). \quad (3)$$

where hidden activation function f is one of three nonlinear functions operating elementwise on matrices; the hyperbolic tangent function (tanh), the logistic sigmoid function, or the rectified linear unit function (ReLU). The trainable parameters are the L hidden weight matrices ($\mathbf{W}$), L bias vectors ($\mathbf{b}$), and the output layer weights and bias $\mathbf{U}$ and $\mathbf{a}$.

All model parameters were trained with the Adam optimization algorithm [13], a variant of stochastic gradient descent. The training loss is the cross-entropy between the true distribution as determined by inconclusive bounds and the DNN's predicted distribution.

The DNN hyper-parameters are model choices which cannot be learned via the training data through gradient descent. They are instead selected by evaluating models on the held out development set which is distinct from the training data and the testing set. The model choices we select in this fashion are the number of hidden layers, the size of each hidden layer (dimensions of the weight matrices $\mathbf{W}$), the hidden activation function, and finally, the mini-batch size and learning rate used in stochastic gradient descent optimization.

We developed and trained our model architecture using the Tensorflow [5] Python library. Due to the small data set and GPU acceleration for computation, it takes under a minute to train a typical model.

Class Labels

As already mentioned, the $\Delta\Delta G$ values in ProTherm – especially those near zero – must be used with caution. To help determine which range of $\Delta\Delta G$ values should delimit stabilizing, destabilizing, and inconclusive mutations, we employed a principled approach by training models across a systematic set of different inconclusive ranges to train the best predictive model.

Class labels are represented as probability distributions over the three classes, i.e. real valued vectors in $\mathbb{R}^3$ that contain non-negative values and sum to one. A label for $\Delta\Delta G$ classification has one element as 1 and the other elements are zero. So, $[1\ \ 0\ \ 0]^T$ corresponds to a $\Delta\Delta G$ score which is negative and outside the range of indeterminacy (a destabilizing mutation), $[0\ \ 1\ \ 0]^T$ corresponds to an inconclusive $\Delta\Delta G$ score inside the range of indeterminacy, and $[0\ \ 0\ \ 1]^T$ corresponds to a $\Delta\Delta G$ score which is positive and outside the range of indeterminacy (a stabilizing mutation). To make a prediction from our model's predicted class distribution, $\mathbf{p}$, we pick the most probable index.

Our ultimate goal is to find a pair of $\Delta\Delta G$ values for which a model can be trained to correctly predict the true labels that those cutoffs would create. For example if the $\Delta\Delta G$ value of an experiment is reported to be -0.8, and our model's $\Delta\Delta G$ cutoffs were -0.6 and 0.7, the true label for that mutation would be destabilizing and a correct prediction from our model would also be destabilizing. For our best model, predictions should match true labels as closely as possible.

Experimental Setup

In order to assess our model's effectiveness at classification for different inconclusive bounds, we trained 100 DNN models with random hyper-parameter configurations (the same configurations were used for all cutoff ranges). We normalized $\Delta\Delta G$ by dividing all values by 10, and executed a triangular grid search of cutoff ranges equivalent to -2.0 to 2.0, stepping by 0.1, in unnormalized $\Delta\Delta G$, for a total of 820 cutoff ranges. All 100 hyper-parameter configurations were assessed for each range, for a total of 8,200 configurations.

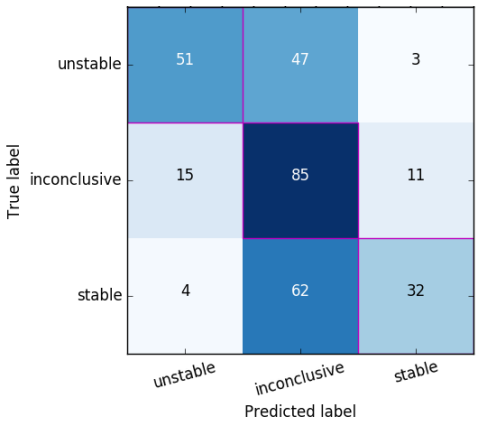


Figure 3: Test set confusion matrix for cutoffs -0.5, 0.5.

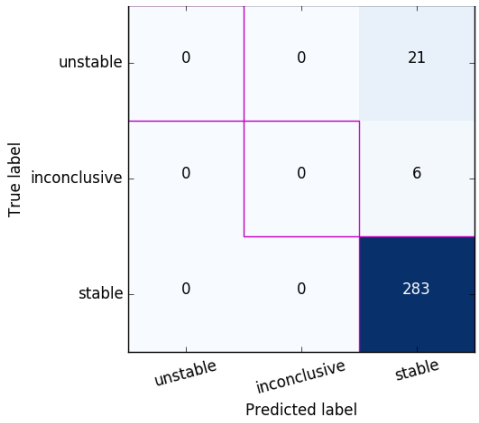


Figure 4: Test set confusion matrix for cutoffs -2.0, -1.9.

Confusion Matrices

For each of the 820 cutoff ranges, we identified the DNN model which achieved the best development set accuracy, and generated a confusion matrix for those model's predictions on the held out test set. Confusion matrices are a method of visualizing the performance of the classification algorithm. They contain the same classes on the vertical and horizontal axis, with the vertical axis indicating true labels for each class and the horizontal axis indicating the model's class predictions. Figure 3 is the confusion matrix generated by the classic heuristic for inconclusive $\Delta\Delta G$ of -0.5 to 0.5. The darker the color the more predictions fall into that intersection of true label and predicted label. A perfect classification model would have predictions only in the top left, center, and bottom right squares.

In addition to the standard metrics of a model's accuracy in predicting the correct class, the confusion matrices offer insights in cases when a model is mis-classified. They allow us to assess Type I and Type II errors, false positive and false negative classifications, and also permit seeing how those incorrect classifications are being classified. This additional information enables assessing whether a particular mis-classification is more detrimental than another. For instance it may be better if a model is less accurate overall, but predicts very few unstable mutations as stabilizing and vice versa, but has a slightly higher than ideal tendency to label mutations as inconclusive.

Results and Discussion

Table 2 reports hyper-parameters as well as several performance metrics for our models. The confusion matrices shown in Figures 3–6 further elucidate these models' performances.

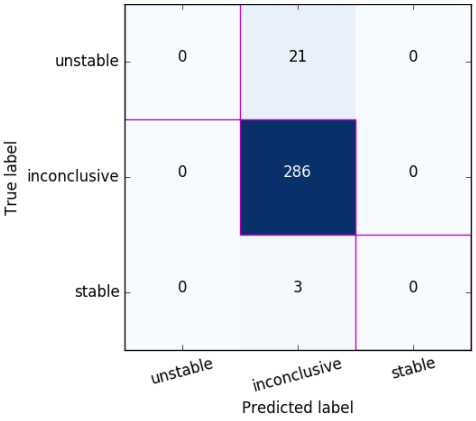


Figure 5: Test set confusion matrix with an unrealistic inconclusive range (-2.0,2.0) where most mutations are labeled as inconclusive.

We first note that when the vast majority of the $\Delta\Delta G$ values fall within a single region determined by the cutoff boundaries, a classification model can trivially achieve high accuracy by learning to predict the majority class. However, labels thus determined may be impractical for scientific pursuits. These situations are characterized by a high proportion of data points which fall into the majority class giving a high majority class accuracy (`macc`), which is indicated in Table 2. One such example is given in Figure 4 which has a small range of indeterminacy, $[-2, -1.9]$, with a large negative offset. For these bounds, `macc` = 91%, with only 6 inconclusive examples and 21 destabilizing examples. We can see from the confusion matrix that all examples were predicted as stabilizing mutations giving a 91% accuracy which amounts to a clearly unhelpful classifier. Another example of ill-conditioned labeling is shown in Figure 5. In this case the indeterminate range is ostensibly too large, $[-2, 2]$ as the model has learned to classify most examples as inconclusive.

	Hyper-parameters					Range		Metrics			
Model	**mb**	**lr**	**hs**	**nl**	**ha**	**L**	**U**	**loss**	**acc**	**macc**	**ratio**
Figure 3	64	0.01	689	1	sigmoid	0.5	0.5	0.96	0.54	0.36	1.51
Figure 4	64	0.01	63	1	sigmoid	-2.00	-1.9	0.29	0.91	0.91	1.00
Figure 5	32	0.09	854	3	ReLU	-2.0	2.0	0.31	0.92	0.92	1.0
Figure 6	64	0.07	361	1	sigmoid	-0.5	0.7	1.15	0.61	0.48	1.28
Figure 7	128	0.01	604	1	sigmoid	-0.4	0.4	0.84	0.60	0.37	1.61

Table 2: Configuration and results for case study models. **mb** denotes minibatch size; **lr** denotes learning rate; **hs** denotes hidden layer size; **nl** denotes number of layers; **ha** denotes hidden activation function; **L** and **U** denote lower and upper cutoff ranges; **loss** denotes average test set cross-entropy between true and predicted values; **acc** denotes accuracy; **macc** denotes majority class accuracy; **ratio** is acc/macc.

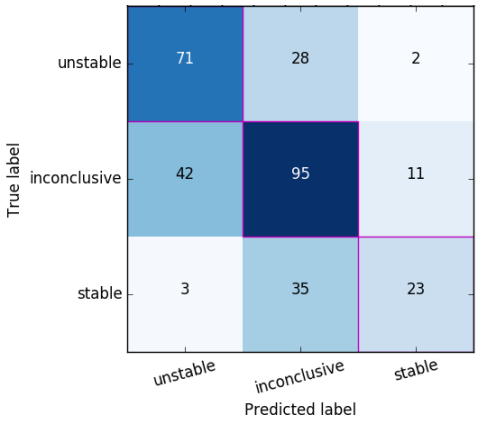


Figure 6: Test set confusion matrix for cutoffs -0.5 and 0.7, where false positive and false negative errors (top-right and bottom-left, respectively) are minimized.

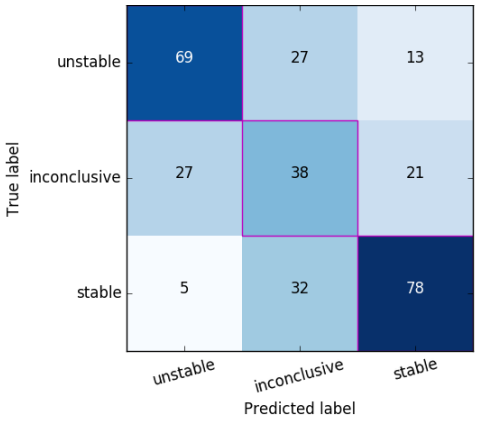


Figure 7: Test set confusion matrix for cutoffs -0.4 and 0.4, where the ratio of accuracy to majority class is maximized.

Figure 3 shows performance for ternary classification using the traditional $\Delta\Delta G$ range for exclusion of examples, $[-0.5, 0.5]$. If we exclude the somewhat innocuous mistakes of examples which are incorrectly classified as inconclusive, along with the examples labeled as inconclusive which would be excluded in the traditional approach in the first place, and attend only to egregious mis-classification of stabilizing as non-stabilizing and vice-versa we achieve a 92.2% accuracy. From this method of preference, running counter to common practice, the optimal ranges for excluding $\Delta\Delta G$ are not necessarily centered on zero.

For instance, based on this criterion of binary predictions within the ternary classification schema, the best cutoff classification range from our experiments is shown in Figure 6 with an inconclusive range $[-0.5, 0.7]$, giving a 94.4% accuracy for the binary subset classification task. On the same test set, for the ternary task, that model achieved an accuracy of 61%. Upon initial assessment this performance does not seem great on its own, but we are more concerned with the model's classification of a destabilizing mutation as a stabilizing one, and vice versa, than we are of it mis-classifying an inconclusive mutation. In this case we see that for this cutoff range the model yields impressive mis-classification rates of 2% for destabilizing to stabilizing and 4% for stabilizing to destabilizing. Such low rates of mis-classification across the inconclusive zone help motivate these findings and suggest that this range is a potentially good $\Delta\Delta G$ cutoff set.

On the other hand, another promising criterion for optimal cutoff is be the ratio of accuracy (`acc`) to majority class accuracy, $\texttt{ratio} = \frac{\texttt{acc}}{\texttt{macc}}$, also displayed in Table 2. For any acceptable model this value should be greater than 1, with larger values being better. Figure 7 shows performance for a model with inconclusive range $[-0.4, 0.4]$ and a significantly higher `ratio` value than the traditional cutoff

Conclusion and Future Work

As an extension on our prior work we were interested in assessing the potential of a deep neural network

for classifying the effects of mutations. We performed a systematic search of the $\Delta\Delta G$ classification cutoff ranges in order to assess the potential viability of a deep neural network ternary classification approach to predicting of mutation affects. Rather than simply accept the general heuristic for classification boundaries of stabilizing, destabilizing or inconclusive, we strove for a more systematic approach. While our findings suggest that the heuristic of -0.5 to 0.5 is not a poor choice by any means, we proposed some compelling arguments for choosing other ranges as boundary conditions for $\Delta\Delta G$ values, namely it is most important to minimize false positive and false negative rates on the ternary task, and maximizing the ratio of accuracy to majority class accuracy are both more important metrics to consider besides accuracy.

For future work we plan to develop robust algorithmic approaches to assess the likely cutoff ranges in ML-based models. We are currently in the development of an end-to-end differentiable approach to jointly learn an optimal cutoff range alongside DNN parameters, as opposed to relying on a parameter sweep as in the current work. Also, expanding our data set with additional mutation $\Delta\Delta$G data – data for proteins with multiple mutations – will likely enhance the DNN's learning and ultimately increase accuracy. We also hope to expand our study into other machine learning algorithms.

Acknowledgments

The authors would like to thank the Nvidia corporation for donating a Titan Xp GPU used in this research.

References

[1] E. Andersson, R. Hsieh, H. Szeto, R. Farhoodi N. Haspel, and F. Jagodzinski. Assessing how multiple mutations affect protein stability using rigid cluster size distributions. In *Proc ICCABS*, pages 1–6, 2016.

[2] E. Andersson and F. Jagodzinski. ProMuteHT : A high throughput compute pipeline for generating protein mutants in silico. In *Proc. CSBW*, 2017.

[3] Emidio Capriotti, Piero Fariselli, and Rita Casadio. I-mutant2. 0: predicting stability changes upon mutation from the protein sequence or structure. *Nucleic acids research*, 33(suppl 2):W306–W310, 2005.

[4] Chi-Wei Chen, Jerome Lin, and Yen-Wei Chu. iStable: off-the-shelf predictor integration for predicting protein stability changes. *BMC bioinformatics*, 14(2):S5, 2013.

[5] Martín Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[6] Roshanak Farhoodi, Max Shelbourne, Rebecca Hsieh, Nurit Haspel, Brian Hutchinson, and Filip Jagodzinski. Predicting the effect of point mutations on protein structural stability. In *Proc. ACM-BCB*, pages 247–252, 2017.

[7] Maria E Figueroa, Omar Abdel-Wahab, Chao Lu, Patrick S Ward, Jay Patel, Alan Shih, Yushan Li, Neha Bhagwat, Aparna Vasanthakumar, Hugo F Fernandez, et al. Leukemic IDH1 and IDH2 mutations result in a hypermethylation phenotype, disrupt TET2 function, and impair hematopoietic differentiation. *Cancer cell*, 18(6):553–567, 2010.

[8] N. Fox, F. Jagodzinski, and I. Streinu. KINARI-Lib: a C++ library for pebble game rigidity analysis of mechanical models. In *Minisymposium on Publicly Available Geometric/Topological Software, Chapel Hill, NC, USA*, June 2012.

[9] Jianglin He, Sunny Choe, Robert Walker, Paola Di Marzio, David O Morgan, and Nathaniel R Landau. Human immunodeficiency virus type 1 viral protein R (Vpr) arrests cells in the G2 phase of the cell cycle by inhibiting p34cdc2 activity. *Journal of virology*, 69(11):6705–6711, 1995.

[10] D.J. Jacobs and B. Hendrickson. An algorithm for two-dimensional rigidity percolation: the pebble game. *Journal of Computational Physics*, 137:346–365, 1997.

[11] D.J. Jacobs, A.J. Rader, M.F. Thorpe, and L.A. Kuhn. Protein flexibility predictions using graph theory. *Proteins 44*, pages 150–165, 2001.

[12] Shuli Kang, Gang Chen, and Gengfu Xiao. Robust prediction of mutation-induced protein stability change by property encoding of amino acids. *Protein Engineering, Design & Selection*, 22(2):75–83, 2008.

[13] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] Tanja Kortemme and David Baker. A simple physical model for binding energy hot spots in protein–protein complexes. *PNAS*, 99(22):14116–14121, 2002.

[15] MD Shaji Kumar, K Abdulla Bava, M Michael Gromiha, Ponraj Prabakaran, Koji Kitajima, Hatsuho Uedaira, and Akinori Sarai. Protherm and pronit: thermodynamic databases for proteins and protein–nucleic acid interactions. *Nucleic acids research*, 34(suppl_1):D204–D206, 2006.

[16] Graham R Serjeant and Beryl E Serjeant. *Sickle cell disease*, volume 3. Oxford university press New York, 1992.

[17] Novel Swine-Origin Influenza A H1N1 Virus and Investigation Team. Emergence of a novel swine-origin influenza a (h1n1) virus in humans. *New England Journal of Medicine*, 2009(360):2605–2615, 2009.

Analysis of Energy Landscapes for Improved Decoy Selection in Template-free Protein Structure Prediction

Nasrin Akhter[1] Amarda Shehu[1,2,3,‡]
[1]Dept. of Computer Science, [2]Dept. of Bioengineering, [3]School of Systems Biology,
George Mason University, Fairfax, VA 22030
[‡]Correspondence: amarda@gmu.edu

Abstract

Decoy selection is the task of automatically extracting near-native structures from an ensemble of low-energy structures generated in silico by a template-free method. Current research shows that discriminating by energy misses near-native structures and allows the inclusion of too many non-native structures. The predominant strategy is to ignore energy and cluster structures by their similarity, offering the top-populated clusters as prediction. In this paper we show that energy can improve accuracy in decoy selection when its inclusion is carried out under the energy landscape view. Specifically, we identify basins in the energy landscape and demonstrate basin selection schemes to outperform clustering. The results are promising and point to further directions of research for improving decoy selection and decoy generation.

1 Introduction

The three-dimensional structure of a protein is central to its biological activities in the cell. The native structure to which the chain of amino acids constituting a protein molecule folds determines to which other molecules the protein sticks and how stably it does so [2]. With protein structure governing recognition events, there is a growing need in the post-genomic era for computational methods to predict native structures of millions of uncharacterized protein-encoding gene sequences [1].

Template-free methods play an important role in determining native protein structures [9]. They operate in the absence of a template structure and generate many low-energy structures (decoys) of a given protein sequence under the umbrella of stochastic optimization. Popular methods include Rosetta [6], Quark [16], and others (e.g, based on evolutionary computation [11, 12]). After generating decoys, one needs to conduct decoy selection; the goal is to automatically extract from the generated ensemble of *decoys*, those that are near-native. In a true blind prediction setting, the true native structure is absent.

Decoy selection remains a challenging problem in computational structural biology. Setting an energy threshold either misses near-native structures or allows the inclusion of too many non-native ones. The predominant strategy is to organize decoys by their structural similarity via clustering. The premise is that structures most similar to others in a dataset are more likely to be drawn from the region containing the native structure. Once clustering has been performed, the top k clusters, with common values of k varying from 1 to 10, are typically offered as the ones possibly containing the unknown native structure [8]. This strategy has varied success, and its utility is tightly related to the quality of decoy generation in the first place [8]. Recent, complementary lines of research on decoy selection utilize machine learning models trained on expert-constructed structural features [10] or discriminate by protein-specific scoring functions [5]. Currently, these lines of research suffer from model transferability.

In this paper we show that energy is an important aspect that can be leveraged to improve accuracy in decoy selection. We show that the inclusion of energy needs to be carried out in the context of the energy landscape probed by the ensemble of decoys generated by a template-free method. Specifically, utilizing spatial data analytics, we identify basins in the energy landscape and propose selection schemes that prove superior to clustering. Our evaluation is stringent and relies not only on the percentage of near-native structures contained in the top k basins or clusters, but on the purity, which operationalizes the idea of minimizing the presence of false positives in what is offered as prediction. The results are promising and point to further directions of research not only for further improving decoy selection but even the computational methods generating decoys in the first place.

2 Method

A molecular energy landscape is a fitness landscape. A fitness landscape consists of a set X of points, a notion $\mathcal{N}(X)$ of neighborhood, distance, or accessibility on X, and a fitness function $f : X \to \mathbb{R}_{\geq 0}$ that assigns a fitness to every $x \in X$. The neighborhood function $\mathcal{N} : X \to \mathcal{P}(X)$ assigns neighbors $\mathcal{N}(x)$ to

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

every $x \in X$. In molecular modeling problems, points $x \in X$ correspond to structures of a molecule, and the fitness function measures the internal energy of a structure. The concept of fitness landscapes originated in theoretical biology more than eighty years ago [15] but has since become a useful construction in many scientific disciplines, from the physics of disordered systems such as spin-glasses, molecular biology [3], characterization of optimization problems in AI [14], and the broader study of complex systems [13].

A fitness landscape can be high-dimensional and multimodal. It may contain many local structures, such as basins and basin-separating barriers. In molecular landscapes, a basin corresponds to a long-lived, thermodynamically-stable or semi-stable structural state of a molecule [3]. The native state of a protein is a basin, though it may not be the deepest one due to artifacts in the current functions designed to measure the internal energy of a protein structure. For this reason, the focus of the rest of the activities is on extracting basins from the landscape probed by a decoy sampling method during decoy generation.

2.1 Elucidating Basins via Graph Embeddings of Landscapes

The notion of a basin is tied to a local, *focal* optimum: a local optimum in the landscape is surrounded by a *basin of attraction*, which is the set of points on the landscape from which steepest descent/ascent converges to that focal optimum. In molecular landscapes, the focus is on local minima. In lieu of observing a molecule rearranging itself (hopping between structures) and reaching a local minimum, one can enrich the landscape with connectivity information to identify focal minima and their basins.

Consider an ensemble of decoys Ω generated by a decoy sampling method. The decoys can be embedded in a nearest-neighbor graph (nngraph) $G = (V, E)$ as follows. The vertex set V is populated with the decoys; that is, $V \leftarrow \Omega$. The edge set E is then populated by inferring the neighborhood structure of the landscape. Given a selected distance function measuring the distance between two decoys, each vertex $u \in V$ is connected to other vertices $v \in V$ if $d(u, v) \leq \epsilon$, with ϵ being a user-defined parameter. A small ϵ may result in a disconnected graph in the presence of non-uniform sampling of the landscape. This can be remedied by increasing ϵ or the number of nearest neighbors of u.

Designing an effective distance function for molecular structures remains an open problem. The root-mean-squared-deviation (RMSD) is commonly used. To remove differences due to rigid-body motions, the RMSD is reported after an optimal superimposition that returns the least RMSD (lRMSD) [7]. To reduce computational costs, the decoys can first be superimposed onto some arbitrarily-chosen reference one, and then only RMSD is used to determine the distance between any two decoys to populate E.

The resulting nngraph can now be analyzed to detect local minima. Let $u \in V$, and let $v \in N(u)$, where $N(u)$ denotes the neighborhood of u. u is a local minimum if $\forall v \in V\ f(u) \leq f(v)$. Each local minimum becomes a focal minimum of some basin. The rest of the vertices are assigned to basins as follows. First, each vertex u is associated a negative gradient estimated by selecting the edge (u, v) that maximizes the ratio $[f(u) - f(v)]/d(u, v)$. Then, from each vertex u that is not a local minimum, the negative gradient is iteratively followed (i.e., the edge that maximizes the above ratio is selected and followed) until a local minimum is reached. Vertices that reach via this process the same local minimum are assigned to the basin associated with that minimum.

2.2 Characteristics of Basins for Selection and Evaluation

For the purpose of identifying the basin that represents the native state among many others possibly revealed by the above analysis, a detailed description is needed in terms of basin characteristics that can help discriminate among basins. We consider three: size, depth, and persistence. Basin size refers to the number of decoys assigned to the same basin. Basin fitness/energy is measured via the fitness/energy of its focal minimum. Basin persistence is a concept used in spatial statistics in the context of filtering out basins attributed to noise in fitness functions [4]. Specifically, the persistence of a basin B is $f(\text{saddle}) - \text{f(B)}$, where $f(B)$ refers to the fitness of B. A (pseudo-)saddle is identified as a vertex u from which the iterative process of following the negative gradient, described above, leads to the focal minimum of B but has a neighbor v with $f(v) < f(u)$ from which the iterative process leads a different local minimum. Effectively, persistence measures how shallow a basin is. In spatial statistics, a persistence threshold p_thresh can be specified as a way of retaining only basins with persistence above p_thresh (merging those with lower persistence into the surviving basins).

For the purpose of evaluation, two more characteristics can be associated with each basin. The first measures the extent to which a basin captures the native state. For a protein with a known native structure, all decoys with lRMSD within a user-set threshold dist_thresh can be deemed to be near-native (thus, populate the native state). Given a basin B, $n(B)$ measures the percentage of near-native decoys in the ensemble Ω that are in B. Effectively, this measures the proportion of true positives. One can trivially increase this by increasing the size of a basin. When

filtering out basins by persistence, smaller basins get merged into larger ones and can trivially increase the proportion of true positives in the larger basins. However, what is more important in a selection strategy is to identify basins with a low number of false positives. For this purpose, we associate a purity p with each basin and define it as the proportion of near-native decoys relative to the size of a basin (thus, penalizing large basins with many false positives).

Implementation Details We use the Structural Bioinformatics Library (SBL) [4] to organize a decoy ensemble Ω into basins. We initialize ϵ to be 1Å, and increase the number of nearest neighbors (per the conformational analysis protocol in SBL) until the nn-graph is connected. For the purpose of analysis, we vary p_thresh $\in [1, 10]$, but uniformly on all test cases, low persistence values $\in [1, 3]$ result in better basins (according to our evaluation metrics).

Experimental Setup Leader clustering is used as the baseline; decoys are shuffled and then either form a new cluster (becoming its representative) or are assigned to the first cluster whose representative is within ϵÅ. The dist_thresh parameter for determination of near-native conformations is set on a per-case basis, as there are difficult proteins on which Rosetta does not get close to 3Å of the native structure. If the lowest lRMSD (over all decoys), which we refer to as min_dist ≤ 1, dist_thresh is set to 2. Otherwise, dist_thresh is set to the minimum value that results in a nonzero number of near-native decoys populating the largest-size cluster. Two basin selection strategies are proposed and compared to leader clustering and each-other. First, basins are sorted by their size, and the top three basins are considered for selection. Alternatively, basins are sorted by size, and the the top ten basins are sorted by their energy; the top three basins in this sorted order are then considered for selection. The evaluation is based on the proportion of true positives (n) and purity (p).

Twelve proteins of different folds and lengths have been selected as test cases, listed in Table 1. They represent easy, medium, and difficult cases for Rosetta. The Rosetta `abinitio` protocol is used to generate decoys for each protein sequence. While Rosetta developers argue for a decoy ensemble to be between 10-20K, we generate around $50,000$ decoys per protein so that dataset size is not an issue for decoy selection.

3 Results

Two sets of results are related. First, we provide visualization of the three top clusters and basins (under each selection strategy) over the generated decoys. Second, we provide a summary comparative analysis that evaluates the three selection strategies.

Table 1: Testing Dataset

PDB ID	Fold	Length	$\|\Omega\|$	min_dist (Å)
1ail	α	70	$53,568$	0.50
1dtdb	$\alpha+\beta$	61	$57,839$	0.51
1wapa	β	68	$51,841$	0.56
1tig	$\alpha+\beta$	88	$52,099$	0.60
1hz6a	$\alpha+\beta$	64	$57,474$	0.72
1c8ca	β	64	$53,322$	1.08
2ci2	$\alpha+\beta$	65	$52,220$	1.21
1bq9	β	53	$53,663$	1.30
1fwp	$\alpha+\beta$	69	$53,133$	1.56
1sap	β	66	$51,209$	1.75
2h5nd	α	123	$51,475$	2.00
1aoy	α	78	$52,218$	3.26

3.1 Visualizing Top Clusters and Basins

Fig 1 selects two test cases and shows decoys generated for each of them as two-dimensional dots, with the x axis tracking the lRMSD of each decoy from the known native structure, and the y axis tracking the Rosetta score12 (all-atom) energy (measured in Rosetta Energy Units – REUs). Decoys belonging to a cluster or a group are colored in red, green, or blue, so as to visualize the top three clusters and basins (under each basin selection strategy).

Fig 1 indicates that the protein with known native structure under PDB id 1dtb is a trivial case for decoy selection, as the top three clusters capture the majority of the near-native conformations. The top three basins under each selection strategy capture similar regions of the decoy space. The protein with known native structure under PDB id 1fwp presents a more difficult case. The top three clusters have many decoys with large lRMSDs from the native (low purity). The purity of the top three basins, when selecting by size or selecting by size and energy, is much higher.

A detailed comparative visualization of the top three basins in each basin selection strategy is shown for four more proteins in Fig. 2. Fig. 2 shows that selecting basins by size and then energy results in the top three basins better discriminating against non-native decoys. Even in cases that seem challenging for basin selection (the proteins with native structure under PDB ids 2ci2 and 1aoy), sorting basins both by size and energy results in purer basins. These graphical results suggest that not ignoring focal energy in the selection scheme may prove promising, and the comprehensive analysis over all test cases, detailed below, supports this conclusions.

3.2 Summary Comparative Analysis

Table 2 compares the three selection strategies as follows. It considers the scenario where G_{1-x} groups

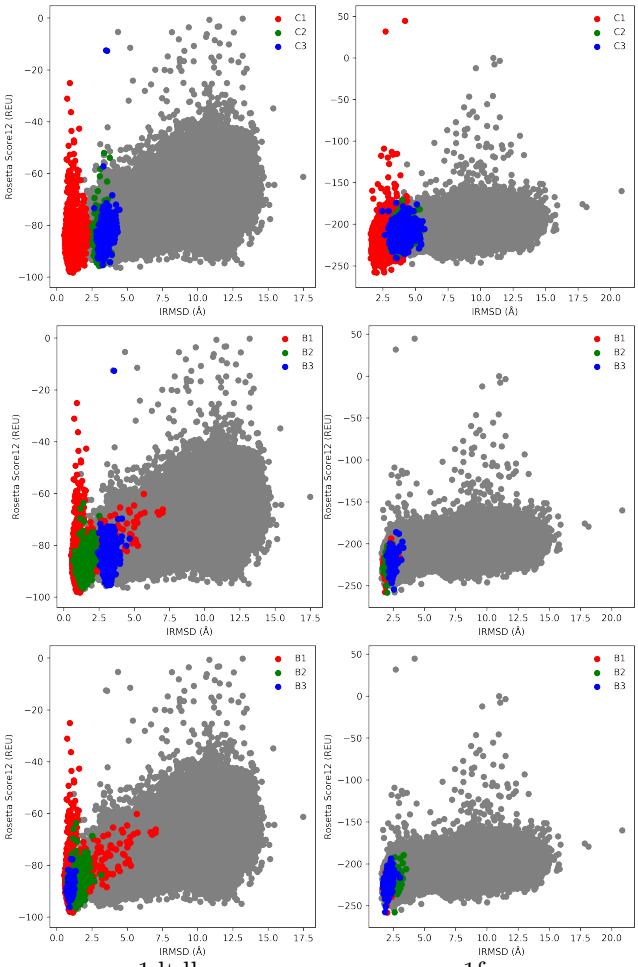


Figure 1: Visualization of the three largest clusters (top panel) versus the three largest basins (middle panel) and the three largest and lowest-energy basins (bottom panel) for two selected proteins with known native structures under PDB ids 1dtdb and 1fwp. Decoys are plotted by their lRMSD from the native structure and their Rosetta score12 all-atom energy.

of decoys are offered as prediction in a decoy selection setting. Based on the selection strategy, a group indicates a cluster or a basin. For instance, C_{1-x} indicates that the top (largest) x clusters are selected and evaluated. Similarly, B_{1-x} indicates that the top x basins (under each ordering, size only, or size and energy) are selected and evaluated. Table 2 limits $x = 3$ and shows the percentage of near-native conformations and the purity in each setting; note that the purity of G_{1-x} is determined by merging all decoys in G_{1-x}. In addition, the relative size of each G_{1-x} (proportional to $|\Omega|$), is shown for reference.

Table 2 allows reaching a few conclusions. First, there are proteins on which a clustering-based selection strategy does well in terms of purity (1dtdb, 1wapa,

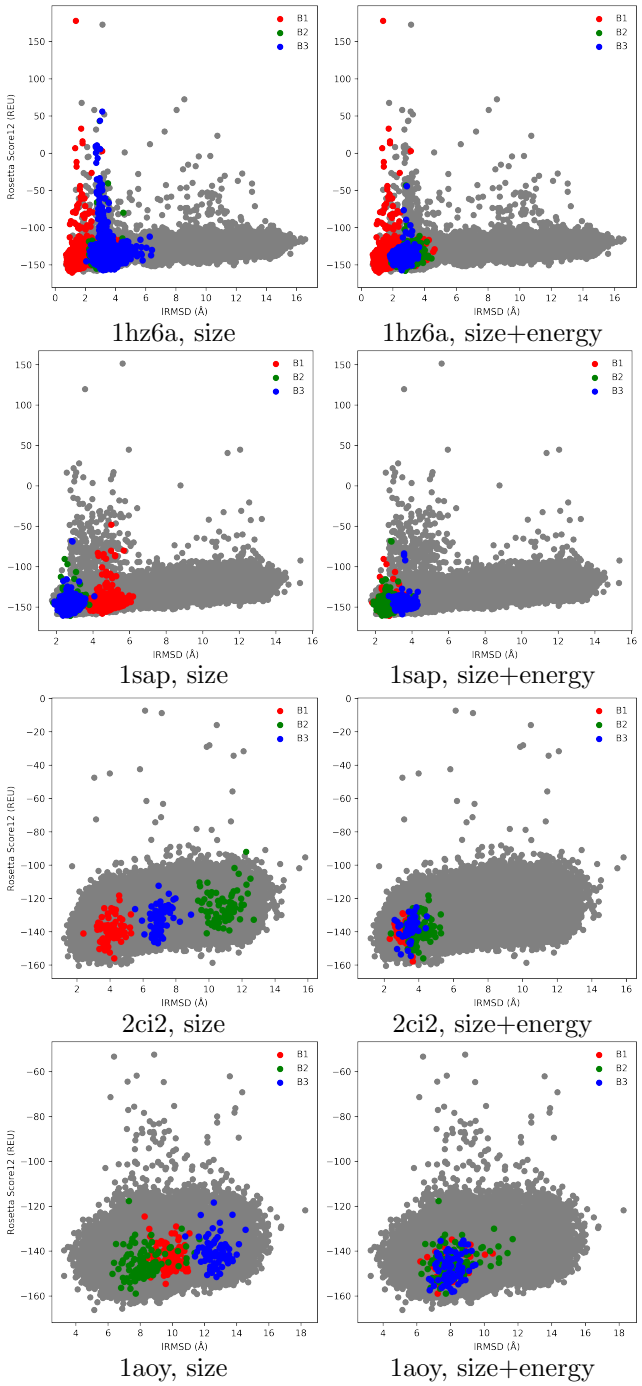


Figure 2: Visualization of the three largest basins (left) and the three largest and lowest-energy basins (right) for four more selected proteins (indicated by the PDB id of their native structure). Decoys are plotted by their lRMSD from the native structure and their Rosetta score12 all-atom energy.

1tig, and 2ci2). On these 4 cases, at least one basin selection strategy does similarly well. On 7/12 cases, clustering is outperformed by both selection strategies in terms of purity (1ail, 1hz6a, 1c8ca, 1bq9, 1fwp, 1sap, 1aoy), particularly when restricting to the top

or top two clusters/basins. By achieving higher purity, the basin selection strategies confer higher likelihood that drawing at random from them will result in a near-native decoy (rather than a false positive). The results in Table 2 also show that considering energy in the basin selection strategy does not result in lower purity; instead, in 8/12 cases, selecting by both size and energy results in higher or same purity over selecting only by size (B_{1-3} in 1ail, 1dtdb, 1c8ca, 2ci2, B_{1-2} and B_{1-3} in 1bq9, B_{1-2} and B_{1-3} in 1hz6a, B_{1-2} and B_{1-3} in 1sap, and B_{1-2} and B_{1-3} in 1aoy).

4 Conclusion

The results presented here suggest that basins in the energy landscape probed by a template-free structure prediction method hold promise for automatically detecting near-native structures of a protein. While energy is often ignored in favor of structural similarity in decoy selection, the presented work indicates that energy can be reliably employed to organize structure data into basins. The results support that selection of basins is more effective than selection of clusters for decoy selection. Considering not just the size but also the energy of a basin is more effective in yielding high-purity basins. This is an important contribution of this paper, as it suggests a landscape-based way of selecting decoys that lowers the number of false positives (non-native decoys) reported. The presented work opens many lines of enquiry regarding probing additional characteristics of basins for selection and extending the analysis to landscapes of reduced dimensionality, as well as beyond structure prediction.

Acknowledgements

This work is supported in part by NSF Grant No. 1421001 and a Jeffress Memorial Trust Award. Computations were run on ARGO, a research computing cluster provided by the Office of Research Computing at George Mason University.

References

[1] C. E. Blaby-Haas and V. de Crécy-Lagard. Mining high-throughput experimental data to link gene and function. *Trends Biotechnol*, 29(4):174–182, 2013.

[2] D. D. Boehr and P. E. Wright. How do proteins interact? *Science*, 320(5882):1429–1430, 2008.

[3] J. D. Bryngelson, J. N. Onuchic, N. D. Socci, and P. G. Wolynes. Funnels, pathways, and the energy landscape of protein folding: a synthesis. *Proteins: Struct. Funct. Genet.*, 21(3):167–195, 1995.

[4] F. Cazals and T. Dreyfus. The structural bioinformatics library: modeling in biomolecular science and beyond. *Bioinformatics*, 33(7):997–1004, 2017.

[5] J. He, J. Zhang, Y. Xu, Y. Shang, and D. Xu. Protein structural model selection based on protein-dependent scoring function. *Statistics and Interface*, 5:109–115, 2012.

[6] A. Leaver-Fay et al. ROSETTA3: an object-oriented software suite for the simulation and design of macromolecules. *Methods Enzymol*, 487:545–574, 2011.

[7] A. D. McLachlan. A mathematical procedure for superimposing atomic coordinates of proteins. *Acta Crystallogr. A.*, 26(6):656–657, 1972.

[8] K. Molloy, S. Saleh, and A. Shehu. Probabilistic search and energy guidance for biased decoy sampling in ab-initio protein structure prediction. *IEEE/ACM Trans. Bioinf. and Comp. Biol.*, 10(5):1162–1175, 2013.

[9] J. Moult, K. Fidelis, A. Kryshtafovych, T. Schwede, and A. Tramontano. Critical assessment of methods of protein structure prediction (CASP) round x. *Proteins: Struct. Funct. Bioinf.*, 82:109–115, 2014.

[10] S. C. Ngan, L. H. Hung, T. Liu, and R. Samudrala. Purely structural protein scoring functions using support vector machine and ensemble learning. *IEEE/ACM Trans Comput Biol*, pages 1–14, 2016.

[11] B. Olson and A. Shehu. Multi-objective stochastic search for sampling local minima in the protein energy surface. In *ACM Conf on Bioinf and Comp Biol (BCB)*, pages 430–439, Washington, D. C., September 2013.

[12] B. Olson and A. Shehu. Multi-objective optimization techniques for conformational sampling in template-free protein structure prediction. In *Intl Conf on Bioinf and Comp Biol (BICoB)*, pages 143–148, Las Vegas, NV, 2014.

[13] S. Samoilenko. Fitness landscapes of complex systems: Insights and implications on managing a conflict environment of organizations. *Complexity & Organization*, 10(4):38–45, 2008.

[14] A. Shehu. Probabilistic search and optimization for protein energy landscapes. In S. Aluru and A. Singh, editors, *Handbook of Computational Molecular Biology*. Chapman & Hall/CRC Computer & Information Science Series, 2013.

[15] S. Wright. The roles of mutation, inbreeding, crossbreeding, and selection in evolution. In *Intl Congress of Genetics*, pages 356–366, 1934.

[16] D. Xu and Y. Zhang. Ab initio protein structure assembly using continuous structure fragments and optimized knowledge-based force field. *Proteins: Struct. Funct. Bioinf.*, 80(7):1715–1735, 2012.

Table 2: Comparison of cluster- and basin-selection strategies: The top G_{1-x} groups of decoys selected from each selection strategy, with x limited to 3, are analyzed. Recall that C_1 refers to the top cluster selected by the cluster-based strategy, whereas B_1 refers to the top basin selected by a basin-based strategy. When analyzing B_{1-x}, the top x basins are merged. The analysis lists the percentage of near-native decoys (n), the purity (p), which is the proportion of near-native decoys relative to the size of a group), and the relative size (s, which is proportional to $|\Omega|$) of each group (cluster or basin).

	Cluster Size			**Basin Size**			**Basin Size + Energy**		
	C_1	C_{1-2}	C_{1-3}	B_1	B_{1-2}	B_{1-3}	B_1	B_{1-2}	B_{1-3}
1ail (0.5Å)	n:12.1% p:10.3% s:7.4%	n:93.2% p:45.2% s:13.1%	n:93.2% p:34.4% s:17.5%	n:47.2% p:100% s:3%	n:48.4% p:52.8% s:5.8%	n:48.4% p:44.8% s:6.9%	n:1.2% p:2.8% s:2.8%	n:48.4% p:52.8% s:5.8%	n:61.9% p:58.6% s:6.7%
1dtdb (0.51Å)	n:99.5% p:99.6% s:22.8%	n:99.5% p:95% s:23.9%	n:99.5% p:91.7% s:24.7%	n:85.3% p:99% s:19.7%	n:95% p:98.9% s:21.9%	n:95% p:94.8% s:22.9%	n:85.3% p:99% s:19.7%	n:95% p:98.9% s:21.9%	n:95.9% p:98.9% s:22.1%
1wapa (0.56Å)	n:98.1% p:98% s:10.2%	n:98.1% p:88.2% s:11.3%	n:98.1% p:81.4% s:12.3%	n:76.8% p:98.9% s:7.9%	n:81.8% p:98.8% s:8.4%	n:86.3% p:98.7% s:8.9%	n:76.8% p:98.9% s:7.9%	n:79.1% p:98.9% s:8.2%	n:84.1% p:98.8% s:8.7%
1tig (0.6Å)	n:50.4% p:94.5% s:7.9%	n:93.5% p:94.8% s:14.9%	n:93.5% p:80.2% s:17.6%	n:28.8% p:100% s:4.4%	n:40.1% p:99.6% s:6.1%	n:50.2% p:99.7% s:7.6%	n:2.7% p:88.4% s:0.5%	n:31.5% p:98.9% s:4.8%	n:42.8% p:98.8% s:6.5%
1hz6a (0.7Å)	n:0.06% p:0.07% s:8.7%	n:9.4% p:7.1% s:15%	n:9.4% p:5% s:21.1%	n:55.5% p:85.5% s:7.3%	n:55.5% p:50% s:12.6%	n:55.5% p:39.3% s:16%	n:55.5% p:85.5% s:7.3%	n:55.6% p:66.6% s:9.4%	n:55.7% p:55.7% s:11.3%
1c8ca (1.1Å)	n:31.3% p:7.8% s:17.8%	n:31.3% p:6.6% s:21%	n:75.7% p:13.9% s:24.2%	n:1.9% p:4.2% s:2%	n:29.3% p:36.1% s:3.6%	n:29.8% p:26.5% s:5%	n:1.7% p:10.1% s:0.8%	n:29.2% p:55.2% s:2.3%	n:32.7% p:53.4% s:2.7%
2ci2 (1.2Å)	n:3% p:100% s:0.7%	n:5.5% p:95.4% s:1.3%	n:7.9% p:95.6% s:1.9%	n:0.3% p:47.2% s:0.1%	n:0.3% p:23.6% s:0.3%	n:0.3% p:15.9% s:0.4%	n:0.4% p:100% s:0.1%	n:0.7% p:68.9% s:0.2%	n:1.1% p:76.9% s:0.3%
1bq9 (1.3Å)	n:0% p:0% s:1.1%	n:0% p:0% s:1.9%	n:0% p:0% s:2.5%	n:60% p:15.5% s:0.2%	n:60% p:7.9% s:0.4%	n:64% p:5.9% s:0.5%	n:60% p:15.5% s:0.2%	n:64% p:9.2% s:0.3%	n:64% p:6.5% s:0.5%
1fwp (1.6Å)	n:99.9% p:20.2% s:3.7%	n:99.9% p:19.3% s:6.1%	n:100% p:18.7% s:7.5%	n:5.6% p:97.7% s:0.3%	n:9.1% p:97.2% s:0.5%	n:10.7% p:84.2% s:0.7%	n:3.5% p:96.4% s:0.2%	n:3.7% p:58.4% s:0.4%	n:9.3% p:77% s:0.7%
1sap (1.75Å)	n:0% p:0% s:12.5%	n:61.9% p:7.2% s:19.9%	n:72.1% p:6.8% s:24.4%	n:0% p:0% s:4.4%	n:32.4% p:9.3% s:8.1%	n:51.4% p:11.5% s:10.3%	n:32.4% p:20.2% s:3.7%	n:51.4% p:20% s:5.9%	n:51.4% p:18.2% s:6.5%
2h5nd (2.0Å)	n:0% p:0% s:0.2%	n:0% p:0% s:0.4%	n:34.2% p:27% s:0.5%	n:0% p:0% s:0.3%	n:0% p:0% s:0.4%	n:16.1% p:13.4% s:0.5%	n:0% p:0% s:0.1%	n:0% p:0% s:0.4%	n:16.1% p:13.7% s:0.5%
1aoy (3.26Å)	n:0% p:0% s:0.9%	n:0.02% p:0.13% s:1.48%	n:0.02% p:0.09% s:2.1%	n:0% p:0% s:0.2%	n:0.2% p:4.9% s:0.4%	n:0.2% p:3.4% s:0.6%	n:0.05% p:3.5% s:0.2%	n:0.23% p:6.9% s:0.4%	n:0.3% p:6.1% s:0.5%

Myocardial Infarction Detection using Multi Biomedical Sensors

Mohammad Mahbubur Rahman Khan Mamun and Ali T Alouani
Department of ECE, Tennessee Technological University
Cookeville, TN, 38505, USA
Mrkhanmamu42@students.tntech.edu

Abstract

Heart failure is one of the diseases that may require frequent physician visit and checkups. Automatic monitoring of specific biomedical signals and using signal analysis techniques, one can assess the patient health condition at his/her own residence and/or work in real time.

In this paper, classification of myocardial infraction condition was diagnosed using measurement from several biomedical sensors by rule based hierarchal decision fusion technique to provide a biomedical heart health assessment technique. The proposed approach combined the progress in signal analysis, sensor data fusion, and rule based simple adaptive threshold decision to process the data in real time and assesses the patient heart condition with low false alarm rate.

Data from ECG, blood pressure (BP) and pulse oximeter (SpO_2) have been used for analysis and diagnosis. Testing using biomedical data form 150 persons were carried out with sensitivity, specificity and accuracy were 94.92%, 92.31% and 93.33% respectively. The Physoinet ECG database was used for evaluation of the methods.

Keywords— Myocardial infarction, data fusion, simple adaptive threshold method, blood pressure, SpO_2.

1 INTRODUCTION

Given the high percentage of current elderly population prone to hypertension and risky heart conditions (such as heart attack), monitoring as well as analyzing real time biomedical signals are needed to avoid unnecessary visits to physician or emergency room. This would save time, money, and the hassle of traveling to a physician due to false alarm conditions.

During myocardial infarction, tissue death due to lack of oxygen can eventually contribute to severe consequences if supply of oxygen is not restored within 90 to 120 minutes [1]. Many researchers studied heart disease classification by improving Electrocardiogram signal analysis. In [2], a method of combining ECG signal from Lead-I and arterial blood pressure to detect premature supraventricular and ventricular contractions (PSC and PVC) which are pre-cursor of serious arrhythmia and other heart diseases. The use of information from one lead gives only heart rate which is not sufficient to detect complex heart diseases. A self-administered functional health infrastructure for data collection and storage using remote monitoring of vital signs such as ECG, blood pressure, respiration, movement, etc. has been proposed in [3]. This work was limited to collection as well as storage of biomedical data and didn't involve processing the data in real time and had to deal with the challenge of data security, storage, and retrieval. With same limitation, [4] used stand-alone medical wireless (or modified to become wireless) devices/sensors to a cell phone using blue tooth communication.

Wavelet transform was used to detect QRS complex (main three deflection in every ECG wave) [5]-[6]. Rothwell et al. showed that the blood pressure variability can be used as an independent variable for strong predictor of ischemic stroke ,even after exclusion of previous stroke patients it provided prediction of myocardial infarction, angina, and heart failure[7] .With the invent of machine learning algorithms, different feature types were used in order to recognize abnormalities in ECG automatically. A common shortcoming of all these approaches is that they are computationally extensive and didn't deal with myocardial infarction detection [8] - [14]. Some recent approaches were based on interval length, amplitude of QRS complex, etc. for pattern recognition but their ability to detect useful patterns decreased when the morphology of the ECG changed [15] - [16]. Currently, most of the methods used to detect potential heart attack scenarios were done by a physician using physical examination (heart rate and chest pain) with ECG or cardiac markers from specific blood tests [17]. Research has been done to perform automatic detection of heart arrhythmia or comparatively simpler heart rhythm related abnormality by analyzing the ECG signal but myocardial infarction detection needs complex algorithm and additional information from complimentary biomedical sensors to perform robust diagnosis decisions.

In this paper, the goal is to use heterogeneous complimentary biomedical sensors to automatically detect symptoms of myocardial infarction. This automated detection system should help the patient monitor his/her heart condition remotely without rushing to hospital when it is not necessary. In available research, arrhythmia or irregular beat detection were done using ECG signal, which were sufficient to provide heart rate. This work goes beyond relying on heart beat detection only. It rather attempts to detect/predict the symptom of heart attack. To accomplish this goal, blood pressure and pulse Oximeter

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

measurements are proposed to complement the information provided by the ECG signals.

2 PROPOSED HEART ATTACK PREDICTION TECHNIQUE

For a potential MI patient the elevation of ST segment (flat isoelectric section of the ECG between the end of the S wave and start of T wave) is one of the first symptom which comes with chest pain [18]. Another sign for MI can be pathological Q wave which once starts to be visible and doesn't go away. The ECG findings of a pathological Q wave include a Q wave with magnitude of > 25 percent of QRS magnitude. High blood pressure (BP) has consistently been associated with an increased risk of MI.Also the control of hypertension with appropriate medication has been shown to reduce the risk of MI significantly [19].To develop a technique that has low probability of false alarms for MI detection, Figure 1 shows a conceptual block diagram of the whole process which includes data acquisition, processing and decision making. Hemodynamic parameters regulating the cardiovascular system are strongly correlated [20].

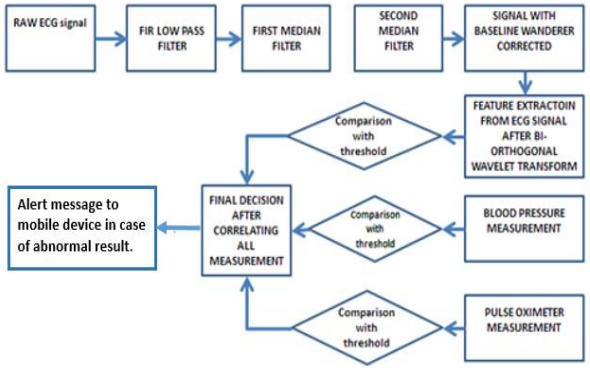


Figure 1: Conceptual Block diagram of the ECG processing and decision recommendation

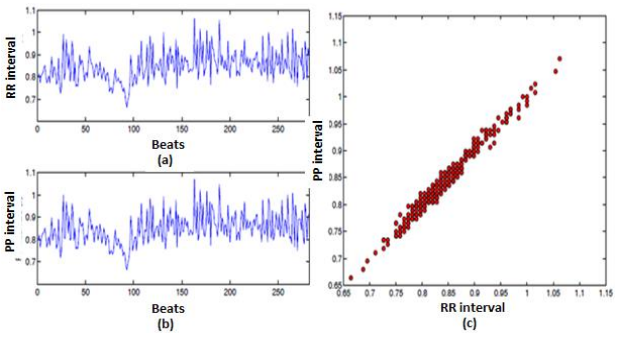


Figure 2: a). RR interval from ECG. b) PP interval from pulse pressure c) correlation between RR interval and PP interval.

Figure2 shows an example of such correlation, where RR interval (the time between two consecutive R waves in ECG measurement) from ECG and pulse pressure interval are present on the left and correlation between those are on the right.

2.1 ECG Sensor and Processing

MIMIC database [21] was used for testing; this database contains multi-parameter recordings obtained from bedside recording of patient. The database includes arterial blood pressure, ECG signal and finger PPG signal with each recording duration is 10 second with average 10 cycle. The systole and diastole are covered 60mmHg to 150 mmHg whereas the ECG was recorded with 500 samples/second with 12-bit resolution [21]. A study using MRI to diagnose myocardial infarction has shown that more emphasis on ST segment depression could greatly improve the yield of the ECG information in the diagnosis of myocardial infarction (sensitivity increase from 50% to 84%) [23]. Pathological Q wave indicates prior or current myocardial infarction, after QT prolongation (measure of the time between the start of the Q wave and the end of the T wave), hyper acute T waves are the earliest-described electrocardiographic sign of acute ischemia, preceding ST-segment elevation [23]. A Matlab GUI was developed for convenience, to determine the symptoms for myocardial infarction which runs through the ECG signal beat by beat and extracts all necessary features.

Maintaining all required criteria, bi-orthogonal wavelet is the most common choice for ECG signal analysis [22] .Using temporal feature conservation ability of bi-orthogonal wavelet transform, the features such as PQRST peak and ST segments can be calculated. Wavelet transform with features such as scale, transition, mother wavelet, orthogonality can preserve both time and frequency domain information at the same time with certain accuracy .The shape of mother wavelet is very significant because there should be maximum correlation between our signal of interest (ECG) and mother wavelet. Using bi-orthogonal wavelet transform, the signal can be decomposed into 4 scales ranging from 2^1 to 2^4. The larger scale relates to lower frequency and smaller scale relates to higher frequency components. Most of the energy of QRS complex is found using 2^3 and 2^4 on the other hand the noises such as electrical interference, muscle activity etc remain in 2^1 and 2^2 level.

2.2 Blood Pressure and Pulse Oximeter System

BP indicates the force of blood through the heart, systolic pressure is when the when the blood ejects from atrium or ventricle and diastole pressure is when atrium or ventricle fills up with blood. On the other hand, the features available in ECG also signify the contraction and expansion of atrium & ventricle. Pressure generated by heart, duration of systole, mean arterial pressure, pulse wave velocity, pulse wave reflection and stiffness of the arterial vessels

influence the blood pressure. So, not only the systole-diastole pressure point but also continuously recorded blood pressure waveform should be analyzed for appropriate representation of cardiac shock. In treated hypertensive participants, the heart rate for systolic blood pressure with potential myocardial infarction and stroke are less pronounced than in participants without treated hypertension [24].Hypertensive heart disease is the leading cause of death from high blood pressure. Hypertension has been shown to be related to risk factors for kidney failure, heart failure, and myocardial infarction and stroke [25].

Pulse Oximeter is a simple and low cost sensor which provides measurement of oxygen level in blood. A percentage over 95 indicates healthy body Oxygen saturation. It can be lower than 93% due to respiratory disease or congenital heart disease. Therefore, monitoring blood saturation can be used as an indication of one of those severe cardiovascular conditions. From pulse Oximeter sensor, the irregular heart bit as well as reduction of oxygen saturation in blood can be observed. Though oxygen saturation is commonly used for monitoring critical patient, in a study [26] baseline SpO2 provided reliable information establishing the diagnosis and severity of acute heart failure as a complication of acute myocardial infarction with a warning cut-off value of <93 percent. The use of pulse oximetry for diagnosis purposes may be recommended when managing patient with risk of acute myocardial infarction [26].

2.3 Decision Fusion from Sensor

A data fusion system must perform whether the data presents different aspect of same event, its redundancy and mismatch. Two mainstream data fusion techniques are, rule-based decision making and fuzzy logic decision making. Here we adopted the rule based approach by taking into account measurements of ECG, blood pressure and SpO_2, these are fused to get more accurate estimation of actual patient parameters and status. Fusion of multimodal event can be modeled as multidimensional process as below:

$$O(m) = [A(m)\ B(m)\ S(m)] \quad (1)$$

Where m denotes the discrete time and A(m) , B(m) and S(m) point to ECG measurement, Blood pressure measurement and SpO_2 level respectively in equation (1).

$$A(m) = (a(m), a(m+1), a(m+2), a(m+3)\ldots\ldots) \quad (2)$$

$$B(m) = (b(m), b(m+1), b(m+2), b(m+3)\ldots\ldots) \quad (3)$$

$$C(m) = (c(m), c(m+1), c(m+2), c(m+3)\ldots\ldots) \quad (4)$$

In equation (2) a(m) presents ECG measurement at $(m)^{th}$ instant of time, b(m) refers blood pressure and c(m) the SpO_2 at m^{th} instant respectively by equation (3) and (4).

$$A(m) = [a_1(m), a_1(m+1), a_1(m+2), \ldots\ldots), a_2(m), a_2(m+1), a_2(m+2), \ldots\ldots)] \quad (5)$$

Here, $a_1(m)$ and $a_2(m)$ are two parameters of ECG features extracted from processed ECG measurements at m^{th} instant in equation (5).

$$B(m) = [b_1(m), b_1(m+1), b_1(m+2), \ldots\ldots), b_2(m), b_2(m+1), b_2(m+2), \ldots\ldots), b_3(m), b_3(m+1), b_3(m+2), \ldots\ldots)] \quad (6)$$

Here, $b_1(m)$, $b_2(m)$ and $b_3(m)$ are systolic , diastolic and mean pressure extracted from patients' blood pressure measurement as in equation (6).

$$C(m) = [c_1(m), c_1(m+1), c_1(m+2), \ldots\ldots)] \quad (7)$$

Here, $c_1(m)$ presents the SpO_2 measurement at m^{th} as in equation (7). Multiple measurements of same data can be fused to yield single estimation which get rid of the erratic measurement which is wayward than the average of other data. Each feature from ECG measurement is analyzed from several windows to use the competitiveness of collected data. Another aspect of fusing is multimodal or integration of overlapping data. In this case, each data presents status of part of the total block. The different sensors also provide complementary measurement. For example, heart rate can be achieved from ECG as well as SpO_2.

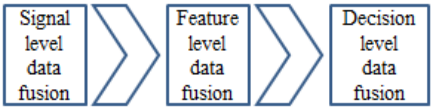


Figure 3: Hierarchical level for data fusion

Three hierarchical levels were used for data fusion as mention in above figure. They are signal level data fusion, feature level data fusion and decision level fusion. Signal level considers the individual signal which provides similar property of an event to deduce parameter. With data which doesn't provide similar property can be used in feature level fusion to come up with a feature vector. Decision level fusion is performed at the top level with either raw data or feature vector to make higher level decision. A rule based decision making implements series of yes/no to check whether a particular condition is existing or not. Our approach is more towards rule based approach and also makes use of the prioritizing aspect of fuzzy logic too. Our objective is to produce an early warning heart attack score (EWHAS) to predict heart attack conditions before a patient is admitted in a hospital. A new index is produced that uses information from only the sensors needed for heart attack prediction such as ECG, BP and oxygen saturation are included.

During MI while the cell death occurs, the ST segment of the ECG gets elevated which is a sufficient diagnosis to start treatment [23]. Pathological Q wave indicates an absence of electrical activity in an area of heart that can be a result of minor myocardial infarction. After QT prolongation, hyper acute T waves are the earliest-described electrocardiographic sign of acute ischemia [26]-[27]. Hypertension, according to Framingham heart study database, is the most common cause of heart failure [28]. Hypertension increases the risk of heart failure four to eight

fold [29]. A study suggests baseline SpO_2 lower than 93% can be considered a sign for acute heart failure (AHF) due to myocardial infarction and the lower the SpO2 value, the higher the probability and severity of AHF [30]. Patients with low (<90%) SaO2 had higher rates of worsening heart failure at 1 to 6 months and also higher rates of mortality, SBP <120 mm Hg was associated with a statistically significant increase in mortality at 1 to 6 months, so combined low SaO2 and systolic blood pressure (SBP) had a particularly strong prognostic implication [31].

Table 1: Feature weight distribution

Features/Measures	Feature weight	Feature weight
abnormal Q wave /Inverted or hyper acute T Wave	1	
abnormal QT interval	2	
ST_Depressions/ST elevation	4	
Hypertension (systolic)	1 (140 -159)	2 (>160)
Hypotension (systolic)	1 (105-90)	2 (< 90)
oxygen saturation	1 (93 to 88)	2 (< 88)

A sensitivity of 50% and specificity of 97% for AMI were achieved with only the currently applied ST-segment elevation criteria while adding the ST-segment depression with elevation increased sensitivity for detection of AMI from 50% to 84% [23]. So the highest weight was given to ST-segment elevation and depression. In a number of epidemiological studies, QT interval prolongation has been associated with an increased risk of being markers of ventricular hypertrophy or myocardial ischemia [32]. This resulted in lower weightage of QT interval. After minor myocardial infarction, people with consistent abnormal Q wave with symptoms such as ST segment change are at 2.7 fold excess risk of coronary death compared to those who have normalized ECG [33]. When an electrocardiogram shows persistent T wave inversion along with ST elevation, further ischemia may make the T wave inversion more pronounced. The lower weight given by the decision system to abnormal Q and T waves reflects their secondary importance when compared to changes in the ST segment [34].

Table 2: Local threshold for ECG features

ECG features	ST elevation	ST depression	hyperacute T wave	pathological Q wave	Prolong Q wave
single feature	counter3/counter1>=0.95	mean_ST_dvalue >1	abs(H_T_peak) >.5	mean_path_c<=-0.25	mean_path_QT >.4
Combination	(counter3/counter1>=0.95 ||mean_ST_dvalue >1) then e=4		(mean_path_c<=-0.25 ||H_T_peak >.5) then f=1		(mean_path_QT >.4) then g=2

High blood pressure increases the likelihood of MI, while excessive drop of blood pressure will hamper coronary perfusion severely introducing new acute coronary events [34].Local decisions of individual sensors are fused by first, for each feature of ECG, care is taken to ensure that an abnormality in a feature appears at least in two consecutive windows of ECG data to avoid false alarm. The final ECG local decision consists of adding the weighted features as shown in table 2. Second, the local decisions made by all the sensors are fused to provide a global and final decision.

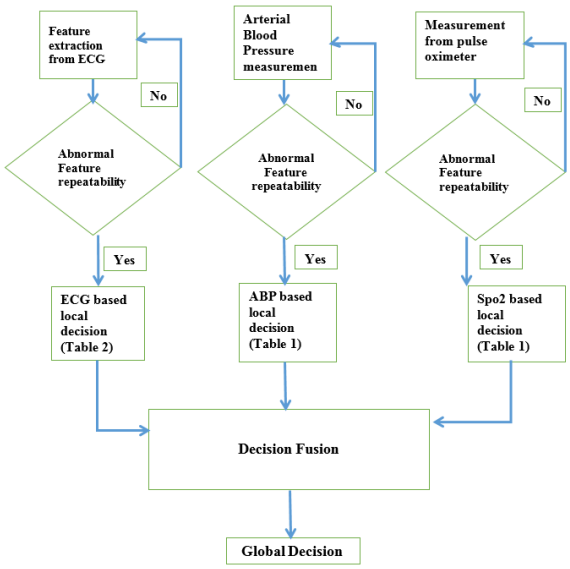


Figure 4: Flowchart for algorithm steps.

The rationale of the global decision uses the fact that prior research reveals ST level is highly correlated with potential heart attacks. However, relying on ST alone will not prevent false alarm from occurring [23]. Table 1 contains the weight assigned to different features used by the decision system. While features from ECG are checked, the weight from extracted features are saved into memory until one finds any relevant reading beyond threshold from blood pressure or oxygen saturation to confirm the potential alert from ECG. Figure 4 provide a flow chart for automatic monitoring and detection of heart health assessment.

3 Biomedical data analysis

Testing of the proposed index system was done using biomedical data from 150 patients from MIMIC database, by extracting necessary features for myocardial infarction. The measurements are classified later using simple adaptive threshold method. A Matlab GUI was developed to process and display relevant features/information forms the different sensors. In the raw ECG wave, the presence of baseline wander and high frequency noise was evident. Two median filters have been used to remove of the baseline wander. The first median filters cancel the prominent QRS complex and second median filter cancel P peak. 2^1 to 2^4 level bi-orthogonal wavelet transform helps to reduce high frequency noise. Understandably 2^4 levels have been picked to extract R peak of ECG signal. To get an idea about iso-scale line (ISO) and ST segment several other points on ECG have to be extracted. P-point with K-point constituted ISO line and J-point with T-point constructed

ST segments. Using the K and P point, the isoelectric line (ISO) can be determined and using J and T point the ST segment can be determined. If 95% of the beat shows ST elevation, a conclusion can be drawn about accurate ST elevation detection. ST depression and pathological Q wave of an ECG signal can be determined.

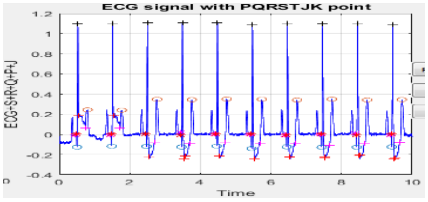


Figure 5: Temporary ST Elevation in first two beats creating a false alarm.

Sometimes, ST elevation or depression or pathological Q wave emerge from ECG signal but it does not stay for long. It is necessary to calculate information from each beat and compare whether the incident is consistent throughout the ECG signal of just a onetime deflection. In figure 5, an example is shown where the ST elevation happened temporarily but it can be concluded as a false symptom of MI.

Table 3: Threshold for decision fusion

MI state	Decision Fusion
Severe case of MI	(e+f+g)>3 && (c_bp>160 || c_bp<90)&& c_ox<88
Mild case of MI	((e+f+g)>0 & (e+f+g)<3)) && (((c_bp >= 140 & c_bp < 160)||(c_bp>=90 & c_bp<105)) || (c_ox>=88 & c_ox<93))
	((e+f+g)>0 & (e+f+g)<3)) && ((c_bp >= 140 & c_bp < 160)||(c_bp>=90 & c_bp<105)) && (c_ox>=88 & c_ox<93)
No MI but arrhythmia symptoms	(H_R>100 || H_R<60)&& (e+f+g)==0
Normal condition	(60<H_R<100) && (e+f+g)==0

The system also takes in to account Blood pressure measurements and Oximeter readings to check whether the readings are normal or abnormal. Such additional information can provide significant insight about the conformity of myocardial infarction.In table 3, the thresholds for decision fusion have been provided. Oxygen saturation decreases with the increase in severity of MI condition and to compensate the ischemic region of heart, heart rate increases as well [28]-[34], on the other hand increase in systolic blood pressure directly relates to risk of MI, stroke and even mortality [24] with the risk getting higher with pressure getting in to different stage of hypertension or hypotension. Thus the threshold values are set similar way to the chronological importance of the event occurs at onset of MI or potential MI. Here, measurement of blood pressure, oximeter are denoted as c_bp and c_ox, respectively. Also features from ECG such as ST elevation/depression, hyper acute T wave and prolonged Q wave are denoted by e, f and g, respectively.

4 Performance evaluation

Four common performance measures have been used to assess the performance of the proposed automatic MI monitoring system: specificity (true negative rate), sensitivity (true positive rate), accuracy, and predictive accuracy.

Table 4: Count of performance measures

Type	TP (True positive)	FN (False negative)	TN (True negative)	FP (False positive)
ECG single feature	46	12	62	30
Fusion of all sensor	56	3	84	7

Table 5: Assessment based on performance measures

Measures	Use multi sensors	Using only ECG	Performance improvement
Sensitivity	94.92%	79.31%	15.60%
Specificity	92.31%	67.39%	24.92%
Accuracy	93.33%	72.00%	21.33%
Predictive value (positive)	88.89%	60.53%	28.36%
Predictive value (negative)	96.55%	83.78%	12.77%

Accuracy measures the probability of correctly diagnosed both diseased and non-diseased persons in the entire population used for testing. Positive predictive value is probability of having positive detection of a diseased person among all the positive result (including false positive result) and negative predictive value is probability of having correct negative detection of a healthy person among all the negative result (including false negative result).A population of 150 persons at rest has been analyzed. The relevant data was obtained from physionet database [21]. The performance of the proposed technique is summarized in table 4 and 5. As shown in Table 5, with only the ECG information, the decision algorithm is prone to false negative and false positive which keeps the sensitivity and specificity lower than acceptable range. However, with the support of blood pressure and oximeter sensor, the false negative and false positive count reduces and sensitivity and specificity improve.

5 Conclusions and Future Work

In this paper, the concept of using multiple complementary biomedical sensors was proposed and applied to MI disease detection. The performance evaluation using 150 patients has shown significant improvement in detecting MI with lower false alarm rate when the proposed technique is used. Similar concept will be used in the future to tackle diseases such as brain stroke.

A pre-requisite for proper use of heterogeneous multi biomedical sensors is the ability to collect all the sensors data at the same time. Toward this goal, a stand-alone

device that collects such data, perform real time diagnosis, and communicate diagnosis results to the appropriate personnel/facility as need is being developed.

References

[1] P.d. Chazal, M. O'Dwyer and R.B. Reilly, "'Automatic classification of heartbeats using ECG morphology and heartbeat interval features," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 7, pp. 1196-1206.

[2] R. Palaniappan and S.M. Krishnan, "'Detection of ectopic heart beats using ECG and blood pressure signals," *2004 International Conference on Signal Processing and Communications, 2004. SPCOM '04*, pp. 573-576.

[3] N.H. Lovell, F. Magrabi, B.G. Celler, K. Huynh and H. Garsden, "'Web-based acquisition, storage, and retrieval of biomedical signals," *IEEE Engineering in Medicine and Biology Magazine*, vol. 20, no. 3, pp. 38-44.

[4] V. GAY and P. LEIJDEKKERS, "'A Health Monitoring System Using Smart Phones and Wearable Sensors," *International Journal of Assistive Robotics and Mechatronics*, vol. 8, no. 2, pp. 29-36.

[5] M. Vaessen, "'A QRS detection method using analog wavelet transform in ECG analysis," *Maastricht University, Department of Mathematics*, vol. 20.

[6] R. Schneider, A. Bauer, P. Barthel and G. Schmidt, "'Challenge 2006: QT interval measurement,", pp. 325-328.

[7] P.M. Rothwell, S.C. Howard, E. Dolan, E. O'Brien, J.E. Dobson, B. Dahlöf, P.S. Sever and N.R. Poulter, "'Prognostic significance of visit-to-visit variability, maximum systolic blood pressure, and episodic hypertension," The Lancet, vol. 375, no. 9718, pp. 895-905.

[8] Mackay, D. (1992). A practical Bayesian framework for backpropagation networks. *Neural Computation*, vol. 4, p. 448-472. 82

[9] HU, Y. H., Palreddy, S., & Tompkins, W. J. (1997). A patient-adaptable ECG beat classifier using amixture of experts approach. *IEEE Transactions on Biomedical Engineering*, vol. 44, p. 891-900.

[10] Coast, D., & Stern, R. (1990). An Approach to Cardiac Arrhythmia Analysis Using Hidden Markov Models, *IEEE Transactions on Biomedical Engineering*, vol. 37, p. 826-836.

[11] Lagerholm, M., Peterson, C., Braccini, G., Edenbrandt, L., & Sornmo, L. (2000). Clustering ECG complexes using hermite functions and self-organizing maps. *IEEE Transactions on Biomedical Engineering*, vol. 47, p. 838-848.

[12] Silipo, R., & Marchesi, C. (1998). Artificial Neural Networks for Automatic ECG Analysis. *IEEE Transactions on Signal Processing*, vol. 46, no. 5, p. 1417-1425.

[13] Thakor, N. V., & Zhu, Y. S. (1991). Applications of cardiac filtering to ECG analysis: Noise cancellation and arrhythmia detection. *IEEE Transactions on Biomedical Engineering*, vol. 39, no. 8, p. 785-794.

[14] Chang, W. H., Lin, K. P., & Tseng, S. Y. (1988). ECG analysis based on Hilbert transform in ECG diagnosis. In Proceedings of *IEEE Engineering and Medical and Biological Society 10th Annual International Conference*, p. 36-37.

[15] Zhou, S. H., Rautaharju, P. M., & Calhoun, H. P. (1993). Selection of a reduced set of parameters for classification of ventricular conduction defects by cluster analysis. In *Proceedings of Computers in Cardiology*, p. 879-882.

[16] Herrero, G. G., Gotchev, A., Christov, I., & Egiazarian, K. (2005). Feature extraction for heartbeat classification using independent component analysis and matching pursuits. In Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 4, p. 725- 728.

[17] A.V. Chobanian, G.L. Bakris, H.R. Black, W.C. Cushman, L.A. Green, J.L. Izzo Jr, D.W. Jones, B.J. Materson, S. Oparil, J.T. Wright Jr, E.J. Roccella, Joint National Committee on Prevention, Detection, Evaluation, and Treatment of High Blood Pressure. National Heart, Lung, and Blood Institute and National High Blood Pressure Education Program Coordinating Committee, "'Seventh report of the Joint National Committee on Prevention, Detection, Evaluation, and Treatment of High Blood Pressure," Hypertension, vol. 42, no. 6, Dec, pp. 1206-1252.

[18] "Coronary Artery Disease - Heart Valve Disease", Heart-valve-surgery.com, 2016. [Online]. Available: http://www.heart-valve-surgery.com/coronary-artery-disease.php. [Accessed: 28- Nov- 2016]

[19] G. Antonakoudis, L. Poulimenos, K. Kifnidis, C. Zouras and H. Antonakoudis, "'Blood pressure control and cardiovascular risk reduction," *Hippokratia*, vol. 11, no. 3, Jul, pp. 114-119.

[20] G. Atanasova and M. Marinov, "'The pulse pressure amplitude as a marker of myocardial infarction risk," Journal of Clinical and Experimental Cardiology, vol. 4, pp. 251.

[21] "PhysioBank ATM", Physionet.org, 2017. [Online]. Available: http://www.physionet.org/cgi-bin/ATM. [Accessed: 29- Apr- 2017]

[22] P. Jouck, "'Application of the Wavelet Transform Modulus Maxima method to T-wave detection in cardiac signals," *Maastricht University Department of Mathematics and Maastricht Instruments,(22)*, pp. 1-32.

[23] T.N. Martin, B.A. Groenning, H.M. Murray, T. Steedman, J.E. Foster, A.T. Elliot, H.J. Dargie, R.H. Selvester, O. Pahlm and G.S. Wagner, "'ST-Segment Deviation Analysis of the Admission 12-Lead Electrocardiogram as an Aid to Early Diagnosis of Acute Myocardial Infarction With a Cardiac Magnetic Resonance Imaging Gold Standard," J.Am.Coll.Cardiol., vol. 50, no. 11, 9/11, pp. 1021-1028.

[24] B.M. Psaty, C.D. Furberg, L.H. Kuller, M. Cushman, P.J. Savage, D. Levine, D.H. O'leary, R.N. Bryan, M. Anderson and T. Lumley, "'Association between blood pressure level and the risk of myocardial infarction, stroke, and total mortality: the cardiovascular health study," Arch.Intern.Med., vol. 161, no. 9, pp. 1183-1192.

[25] S.D. Pierdomenico, M. Di Nicola, A.L. Esposito, R. Di Mascio, E. Ballone, D. Lapenna and F. Cuccurullo, "'Prognostic value of different indices of blood pressure variability in hypertensive patients," Am.J.Hypertens., vol. 22, no. 8, Aug, pp. 842-847.

[26] J. Masip, M. Gayà, J. Páez, A. Betbesé, F. Vecilla, R. Manresa and P. Ruíz, "'Pulse oximetry in the diagnosis of acute heart failure," Revista Española de Cardiología (English Edition), vol. 65, no. 10, pp. 879-884.

[27] "National Early Warning Score (NEWS) - MDCalc", Mdcalc.com, 2017. [Online]. Available: https://www.mdcalc.com/national-early-warning-score-news. [Accessed: 29- Apr- 2017]

[28] G. Lip, D. Felmeden, F. Li-Saw-Hee and D. Beevers, "'Hypertensive heart disease. A complex syndrome or a hypertensive 'cardiomyopathy'?" *Eur.Heart J.*, vol. 21, no. 20, pp. 1653-1665.

[29] W.B. Kannel, D. Levy and L.A. Cupples, "'Left ventricular hypertrophy and risk of cardiac failure: insights from the Framingham Study." *J.Cardiovasc.Pharmacol.*, vol. 10, pp. S135-S140.

[30] J. Masip, M. Gayà, J. Páez, A. Betbesé, F. Vecilla, R. Manresa and P. Ruíz, "'Pulse oximetry in the diagnosis of acute heart failure," *Revista Española de Cardiología (English Edition)*, vol. 65, no. 10, pp. 879-884.

[31] O. Milo-Cotter, G. Cotter, E. Kaluski, M.M. Rund, G.M. Felker, K.F. Adams, C.M. O'Connor and B.D. Weatherley, "'Rapid clinical assessment of patients with acute heart failure: first blood pressure and oxygen saturation--is that all we need?" *Cardiology*, vol. 114, no. 1, pp. 75-82.

[32] J.J. Candil and C.M. Luengo, "'QT interval and acute myocardial ischemia: past promises, new evidences," *Revista Española de Cardiología*, vol. 61, no. 06, pp. 561-563.

[33] N.D. Wong, D. Levy and W.B. Kannel, "'Prognostic significance of the electrocardiogram after Q wave myocardial infarction. The Framingham Study," *Circulation*, vol. 81, no. 3, Mar, pp. 780-789.

[34] K. Channer and F. Morris, "'ABC of clinical electrocardiography: Myocardial ischaemia," *BMJ*, vol. 324, no. 7344, Apr 27, pp. 1023-1026.

Extracting Co-mention Features from Biomedical Literature for Automated Protein Phenotype Prediction using PHENOstruct

Morteza Pourreza Shahri and Indika Kahanda
Gianforte School of Computing, Montana State University
Bozeman, MT 59717, USA
(mpourrezashahri, indika.kahanda)@montana.edu

Abstract

Human Phenotype Ontology (HPO) is a recently introduced standard vocabulary for describing disease-related phenotypic abnormalities in human. Since experimental determination of HPO categories for human proteins is a highly resource-consuming task, developing automated tools that can accurately predict HPO categories has gained interest recently. In our previous work, we developed PHENOstruct, an automated phenotype prediction tool that uses input features generated from heterogeneous data sources including standard bag-of-words features extracted from biomedical literature. In this work, we introduce novel co-mention features which are based on co-occurrences of protein names and HPO terms within a specified span of text. Our experimental results indicate that utilizing co-mentions significantly improves the overall performance and that the most effective span is the paragraph-level. This is the first study that uses a knowledge-based approach for generating literature features for the task of automated protein phenotype prediction. These findings have implications for practitioners interested in developing automated biocuration pipelines for phenotypes.

1 Introduction

Phenotypes can be described as any observable characteristics of an organism which have fascinated researchers' interests since the relationship between a gene and its phenotypic manifestation was discovered [15]. The Human Phenotype Ontology (HPO) provides a bioinformatics resource which offers a framework for the analysis of phenotypic abnormalities associated with human disease [11]. HPO was originally populated based on databases, such as OMIM (Online Mendelian Inheritance in Man) [6], which contain information about rare diseases. Each single protein is linked to a set of HPO terms based on the diseases caused by mutation to the corresponding genes. Currently, only a small portion of human proteins (about 3,500) have HPO annotations, and researchers believe there are more genes related to human diseases (P. Robinson, personal communication, July 12, 2014). Manually annotating proteins with HPO categories through wet-lab experiments and/or clinical studies is a highly-resource-consuming task, and over the last few years there has been a growing interest in developing automated tools to predict protein-HPO term annotations [10, 12, 13, 19]. In fact, automated protein-HPO term prediction was one of the tasks in the recent CAFA challenge [7].

The HPO is composed of independent sub-ontologies that describe various aspects of phenotypes [11]. The main sub-ontology is *Phenotypic abnormality* and it describes clinical abnormalities. The *Mode of inheritance* sub-ontology describes phenotypes according to inheritance patterns and contains terms such as *Autosomal dominant*. The *Mortality/Aging* sub-ontology similarly describes the age of death and contains terms such as *Neonatal death* or *Sudden death*. Finally, the *Clinical modifier* sub-ontology is composed of terms such as *Incomplete penetrance*, and describes typical modifiers of clinical symptoms [11]. Throughout this paper, we use the terms Organ, Inheritance, and Onset, for referring to the Phenotypic abnormality, Mode of inheritance and Clinical modifier, respectively. Within each sub-ontology, categories are arranged in a Directed Acyclic Graph (DAG) structure.

In our previous work, we developed PHENOstruct [10], which is the first computational method for automated prediction of protein-HPO terms. It uses a Structured Support Vector Machine (SSVM) model for predicting hierarchically consistent HPO labels. PHENOstruct employs several heterogeneous data sources as input: protein-protein interactions, disease variants, experimentally validated functional annotations, and biomedical literature, and uses protein-HPO term annotations extracted from the HPO website as the class labels. PHENOstruct used simple Bag-of-Words (BoW) features obtained from the biomedical literature in which all words occurring in the sentences that contain protein names are used as

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

features. However, this is a knowledge-free approach in which information on actual phenotypes mentioned in the literature is not utilized.

Over the last few years, there have been several other automated protein-HPO term prediction tools [10, 12, 13, 19]. Notaro et al. [13] proposed a two-step method that consists of a flat learning in the first step and a hierarchical combination of the predictions in the second step. Valentini et. al. presented a novel Hierarchical Top-Down algorithm that assigns a single classifier to each HPO term and based on the hierarchical structure of DAG, it can correct the predictions [19]. Moreover, Notaro et al. proposed an algorithm that exploits the information from the ontology terms which specifies the phenotype information related to each human gene [12]. However, none of these methods use literature as their input while PHENOstruct extracts literature features using a knowledge-free approach (i.e. BoW) as opposed to a knowledge-based approach in which phenotype information is also considered.

Undoubtedly, the most comprehensive resource on biological findings, including disease-related phenotypes, is the biomedical literature. Therefore, extracting bio-entities from literature and linking them to bio-ontologies such as HPO has attracted interest within the text mining community recently [5]. This approach has high potential for exploiting the data from a variety of patient reports, case studies, and controlled trials [5]. In a related study, GOstruct 2.0 [9] utilized a natural language processing (NLP) pipeline to successfully exploit information on protein function (i.e. Gene Ontology or GO terms [1]) from the literature. Similarly, Funk et. al. [4] conducted a comprehensive study on evaluating the usage of the literature feature for the task of protein-GO term prediction. They, in addition to simple BoW features, extracted protein names and GO terms co-mentions (co-occurrences of protein names and GO terms within a short span of text) from biomedical literature, and demonstrated the utility of a knowledge-based approach for that task. Furthermore, in our previous work that uses the BoW model with PHENOstruct, we found that the majority of the most important tokens extracted from literature (i.e. the tokens that were assigned the highest weights in the trained SSVM) consist of names of proteins, genes, and diseases [10]. This suggested that applying a knowledge-based approach in extracting features would be more effective for phenotype prediction. Moreover, co-mentions have the added value that they are easy to verify by a human curator [10].

In this work, we conduct the most comprehensive evaluation of extracting literature features for the task of protein phenotype prediction. We use a knowledge-based approach and extract protein-phenotype co-mentions (co-occurrences of protein names and HPO terms within a specified span of text) from an extremely large collection of biomedical literature. Using the various co-mention features as input to PHENOstruct, we demonstrate the utility of this approach for the task of automated protein-HPO term prediction. Outcomes of this study have implications for the bio-curation community as well as text mining practitioners interested in utilizing literature for protein phenotype prediction.

The rest of the paper is organized as follows: Section 2 describes the co-mention features, the text mining pipeline used for obtaining the said features as well as the experimental setup, section 3 discusses the key observations from the experiments, and Section 4 presents conclusions and future directions.

2 Methodology

2.1 Data

In this work, we use CAFA3 Targets (http://biofunctionprediction.org) as the reference set of input proteins. We use features generated from protein-protein interactions (downloaded on 09-26-17), disease variants (downloaded on 07-05-17), experimentally validated GO annotations (downloaded on 07-21-17), as well as simple BoW features generated from biomedical literature as input for PHENOstruct (as described elsewhere [10]). Combination of variants, protein-protein iterations and GO features is referred to as VNG. In addition to the simple BoW features, we introduce novel protein-HPO term co-mention features as described below. We use UniProt synonyms of proteins to improve the coverage when extracting protein names from literature. In terms of labels, we use protein-HPO term annotations extracted from the HPO website on 07-18-17. Table 1 depicts the statistics on the HPO labels used for this study. We ignored *Mortality/Aging* sub-ontology in our experiments.

Table 1: Number of proteins, HPO categories, and annotations.

Sub-ontology	Proteins	HPO categories	Annotations
Organ	3,407	2,872	291k
Inheritance	3,049	15	10.3k
Onset	1,053	20	4.9k

2.1.1 Literature Features

We employed 27 million Medline abstracts and 1.6 million full text articles for obtaining the literature features. We generated two different sets of literature

Table 2: Statistics of co-mentions extracted from both Medline and PubMed

Organ				
Span	Unique proteins	Unique HPO terms	Unique co-mentions	Total co-mentions
Sentence-level	2,306	2,475	102,726	1,962,332
Paragraph-level	2,348	2,475	157152	7,292,398
Non-sentence-level	2,181	2,475	137,486	5,423,845
Inheritance				
Span	Unique proteins	Unique HPO terms	Unique co-mentions	Total co-mentions
Sentence-level	1,710	12	5,029	100,086
Paragraph-level	1,763	12	5,930	370,656
Non-sentence-level	1,496	12	4,929	283,740
Onset				
Span	Unique proteins	Unique HPO terms	Unique co-mentions	Total co-mentions
Sentence-level	399	16	1,126	14,965
Paragraph-level	511	16	1,948	74,602
Non-sentence-level	493	16	1,811	59,886

features: (1) Simple bag-of-words (BoW) features [10], (2) co-mention features. Details of the feature generation is described below.

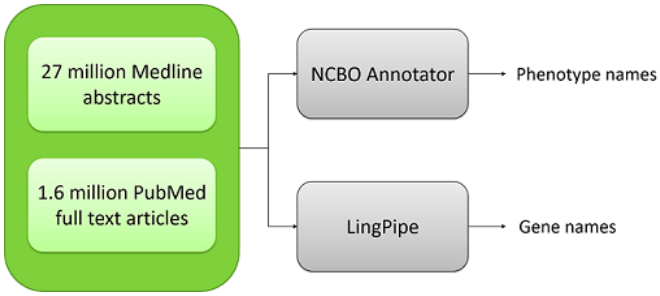


Figure 1: Overview of the NLP pipeline for extracting features from the literature.

Bag-of-words Features

Bag-of-words (BoW) is a knowledge-free feature representation which is broadly used in text mining applications. We retrieved all the sentences which had an occurrence of a protein name as candidates and extracted all the words in those sentences. For each sentence, we first lowercased all the words in the sentence and then removed the highly frequent words (stop words). All the remaining words and their counts were used as feature-value pairs. A protein is represented as a vector of variables, each of which is the count of that specific word.

Co-mentions Features

In this work, we introduce the novel protein-HPO co-mention (CoM) features which are computed from co-occurrences of the protein names and HPO terms within a specified text span. Three text spans were considered for obtaining co-mentions: sentence-level, non-sentence-level and paragraph-level. Sentence-level co-mentions (SCoM) occur in a single sentence and paragraph-level co-mentions (PCoM) are proteins and HPO terms which occur in a single paragraph (i.e. across multiple sentences). Non-sentence-level co-mentions (NSCoM) are obtained by subtracting SCoMs from PCoMs. Note that SCoMs and NSCoMs are proper subsets of PCoMs. Each protein is represented by a vector in which all the HPO terms co-occurred with the protein and their counts specify the feature-value pairs. Statistics on these co-mention features are given in Table 2.

Text Mining Pipeline

We developed the NLP pipeline shown in Figure 1 in order to obtain the BoW and CoM features. We used NCBO Virtual Appliance from BioPortal [8, 14] to extract all the phenotype names from the literature. Protein name mentions were retrieved from the literature files using LingPipe [3] trained on GeneTag [18]. In our preliminary studies, we also considered other alternatives to extract these entities such as OBO annotator [17] and Bio-Lark CR [5] for extracting phenotype names and GNormPlus [20] and ABNER [16] for extracting protein names. However, most of these systems were either difficult to access or did not provide desirable results.

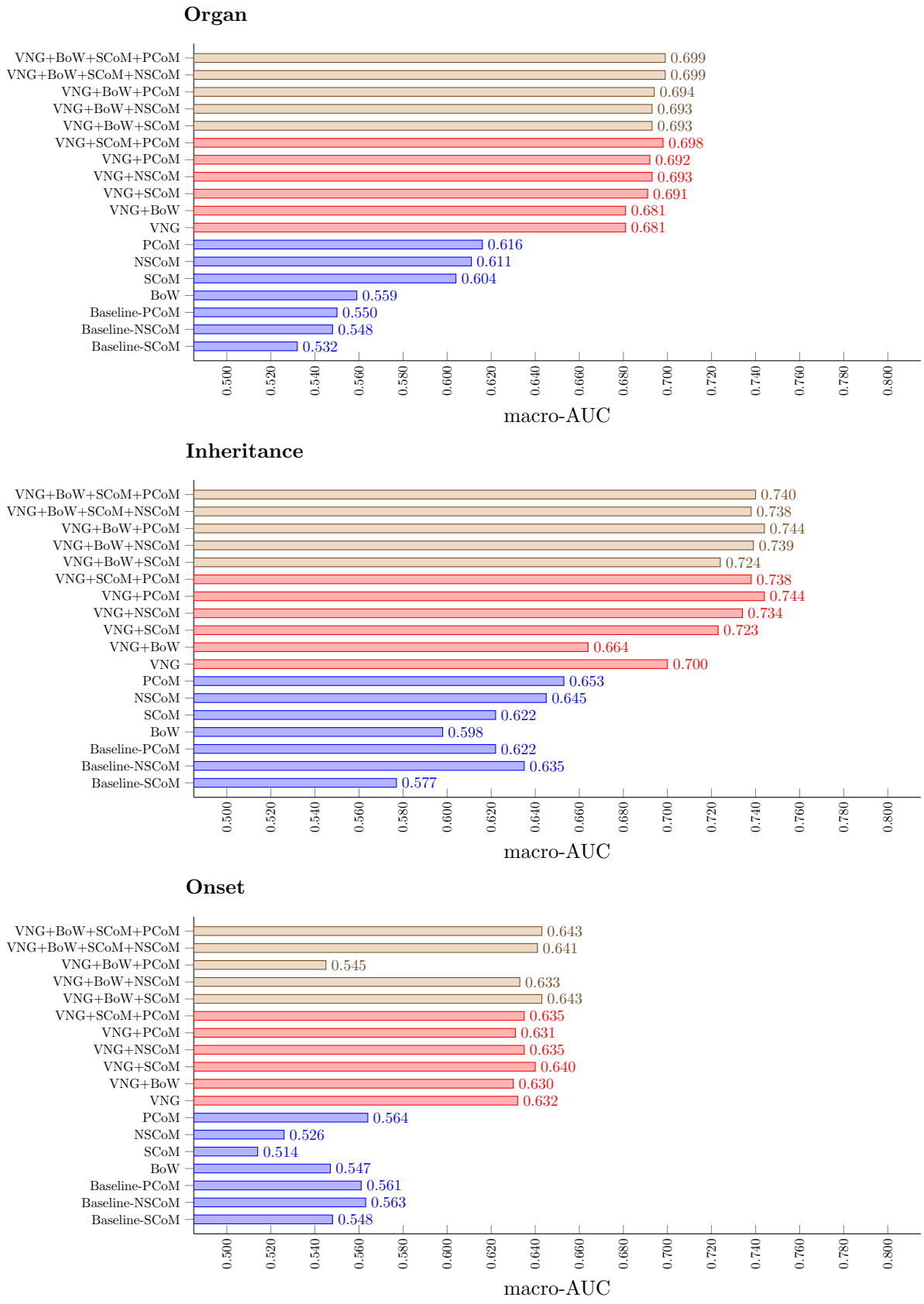


Figure 2: PHENOstruct's performance with different combinations of data sources (VNG: Variants+Network+GO).

2.2 PHENOstruct

As previously mentioned, PHENOstruct is the first computational method for automated protein phenotype prediction problem [10]. It can capture information from the inter-relationships between the HPO labels and has the advantage of not having to train multiple classifiers. Moreover, predicted labels are hierarchically consistent. PHENOstruct employs a Structured SVM model for HPO term prediction. For each test protein provided to the trained model, it outputs a set of HPO labels and corresponding confidence scores.

2.3 Experimental Setup

In order to establish baselines, we utilized the SCoMs, NSCoMs, and PCoMs as the final predictions themselves (i.e. without any machine learning) along with their associated frequencies as the confidence scores. As mentioned before, PHENOstruct provides confidence scores for each prediction. Considering the structure of HPO, all HPO annotations and predictions are expanded via the *true path rule* to the root node of HPO. This rule states that any annotation to a certain term implicitly indicates annotations to all its ancestors. Macro-AUROC (Area Under the Receiver Operative Curve) was used as the evaluation measure for the predictions. We use a five-fold cross-validation setting for all our experiments. Separate experiments were carried out for each sub-ontology. In order to compare the experiments, we compute p-values using paired t-tests by considering only the leaves. All the experiments were performed on a system running Linux Fedora 26 with a 24-cores processor and 128 GB of memory. The average running time for each experiment on the Organ, Inheritance, and Onset sub-ontologies are 36 hours, 20 minutes, and 5 minutes, respectively. Note that PHENOstruct is not compared to other protein-phenotype prediction tools [10, 12, 13, 19] because of the availability of only research-quality code.

3 Results and Discussion

In order to evaluate the effectiveness of the newly introduced co-mention features, a set of ablation studies were carried out by feeding different combinations of features into PHENOstruct as input. Our experimental results demonstrate that, when using individual co-mention features as the input, the paragraph-level co-mentions (PCoMs) provide the best performance in all three sub-ontologies (see Figure 2). PCoMs consistently beat SCoMs and NSCoMs for all three sub-ontologies. These observations suggest that the paragraph level is the span that is overall better if you are interested in using a single set of co-mention features. Bada et al. [2] describes that co-mentions do not necessarily occur in the same sentence; this may be the justification for the relatively superior performance of PCoMs.

Moreover, PCoMs by themselves consistently outperform BoWs in all three sub-ontologies suggesting that knowledge-based approaches are better than knowledge-free methods (P-values for Organ, Inheritance, and Onset are 7.8E-31, 7.6E-01, and 6.2E-01, respectively). This observation is also true for both SCoMs and NSCoMs in many of the cases. Moreover, co-mention features combined with other data sources give better performance compared to using BoW features combined with other data sources.

Another key observation is that, as expected, SCoMs, NSCoMs, and PCoMs outperform the baseline-SCoM, baseline-NSCoM, and baseline-PCoM, respectively, in both Organ and Inheritance sub-ontologies. However, this is not the case in Onset sub-ontology; further investigation is required for finding the underlying reason.

Literature features by themselves provide lower performance compared to when using them in conjunction with other types of data sources. However, the best performance in all three sub-ontologies is obtained by using literature features along with other data sources (P-values for Organ, Inheritance, and Onset are 1.7E-61, 1.4E-01, and 2.4E-02, respectively). This suggests literature features are highly complementary to other data sources.

Regardless of the data sources/features used, aligning with our previous experimental observations [10], PHENOstruct provides best performance in the Inheritance sub-ontology closely followed by the Organ sub-ontology.

4 Conclusion and Future Work

In this work we conducted a comprehensive study on evaluating a variety of literature features for the task of protein phenotype prediction using PHENOstruct. We demonstrate that knowledge-based features (i.e. co-mention features) helps improve the overall performance and are more effective than their knowledge-free counterparts. Moreover, we find that paragraph span best serves this purpose; PCoMs are the most valuable source of information in comparison with the other literature feature sets. However, we conclude that using PCoMs as an individual data source does not provide the best performance, and it needs to be used as a complementary set of features to obtain the most optimum performance.

This study opens up many other avenues for future investigation. By carefully analyzing the the performances of baselines, we notice that our NLP pipeline is generating a large number of false positives (data not shown). In other words, not every co-occurrence of a protein and a phenotype represent a valid relationship. Since our current work does not consider the context surrounding these entity words, the next step would be to develop a context-sensitive co-mention filter/classifier for removing these false positives and improving the overall quality of generated co-mentions. Moreover, this classifier by itself can serve as an important component in a fully automated biocuration pipeline for phenotypes.

References

[1] Michael Ashburner et al. Gene Ontology: tool for the unification of biology. *Nature genetics*, 25(1):25–29, 2000.

[2] Michael Bada, Dmitry Sitnikov, Judith A Blake, and Lawrence E Hunter. Occurrence of gene ontology, protein ontology, and ncbi taxonomy concepts in text toward automatic gene ontology annotation of genes and gene products. *BioLink–an ISMB Special Interest Group. Berlin, Germany: Proceedings of BioLINK SIG*, 2013:13–19, 2013.

[3] Bob Carpenter. LingPipe for 99.99% recall of gene mentions. In *Proceedings of the Second BioCreative Challenge Evaluation Workshop*, volume 23, pages 307–309, 2007.

[4] Christopher S Funk, Indika Kahanda, Asa Ben-Hur, and Karin M Verspoor. Evaluating a variety of text-mined features for automatic protein function prediction with GOstruct. *Journal of biomedical semantics*, 6(1):9, 2015.

[5] Tudor Groza et al. Automatic concept recognition using the Human Phenotype Ontology reference and test suite corpora. *Database*, 2015:bav005, 2015.

[6] Ada Hamosh, Alan F Scott, Joanna S Amberger, Carol A Bocchini, and Victor A McKusick. Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders. *Nucleic acids research*, 33(suppl_1):D514–D517, 2005.

[7] Yuxiang Jiang et al. An expanded evaluation of protein function prediction methods shows an improvement in accuracy. *Genome biology*, 17(1):184, 2016.

[8] Clement Jonquet, Nigam H Shah, and Mark A Musen. The open biomedical annotator. *Summit on translational bioinformatics*, 2009:56, 2009.

[9] Indika Kahanda and Asa Ben-Hur. GOstruct 2.0: Automated Protein Function Prediction for Annotated Proteins. In *BCB*, 2017.

[10] Indika Kahanda, Christopher Funk, Karin Verspoor, and Asa Ben-Hur. PHENOstruct: Prediction of human phenotype ontology terms using heterogeneous data sources. *F1000Research*, 2015.

[11] Sebastian Köhler et al. The Human Phenotype Ontology in 2017. *Nucleic Acids Research*, 45(D1):D865–D876, 2017.

[12] Marco Notaro et al. Ensembling Descendant Term Classifiers to Improve Gene–Abnormal Phenotype Predictions. *Proceedings of CIBB*, page 1, 2017.

[13] Marco Notaro, Max Schubach, Peter N Robinson, and Giorgio Valentini. Prediction of Human Phenotype Ontology terms by means of hierarchical ensemble methods. *BMC bioinformatics*, 18(1):449, 2017.

[14] Natalya F Noy et al. Bioportal: ontologies and integrated data resources at the click of a mouse. *Nucleic acids research*, 37(suppl_2):W170–W173, 2009.

[15] Anika Oellrich et al. The digital revolution in phenotyping. *Briefings in bioinformatics*, 17(5):819–830, 2015.

[16] Burr Settles. ABNER: an open source tool for automatically tagging genes, proteins and other entity names in text. *Bioinformatics*, 21(14):3191–3192, 2005.

[17] Maria Taboada, Hadriana Rodríguez, Diego Martínez, María Pardo, and María Jesús Sobrido. Automated semantic annotation of rare disease cases: a case study. *Database*, 2014, 2014.

[18] Lorraine Tanabe, Natalie Xie, Lynne H Thom, Wayne Matten, and W John Wilbur. GENETAG: a tagged corpus for gene/protein named entity recognition. *BMC bioinformatics*, 6(1):S3, 2005.

[19] Giorgio Valentini et al. Prediction of Human Gene-Phenotype Associations by Exploiting the Hierarchical Structure of the Human Phenotype Ontology. In *IWBBIO (1)*, pages 66–77, 2015.

[20] Chih-Hsuan Wei, Hung-Yu Kao, and Zhiyong Lu. GNormPlus: an integrative approach for tagging genes, gene families, and protein domains. *BioMed research international*, 2015, 2015.

Dynamics of Hepatitis C Virus Infection

Fathalla A. Rihan[1*], Bassel F. Rihan[2]

[1]Department of Mathematical Sciences, College of Science, UAE University, Al-Ain 15551, UAE

[2]Faculty of Medicine, Ain Shams University, El-Abaseya, 11566, Cairo, Egypt

Abstract

Herein, we provide a mathematical model to investigate the dynamics of Hepatitis-C Virus (HCV) replication, in presence of interferon-α (IFN) treatment. We consider a fractional-order in the model to represent the intermediate cellular interactions and intracellular delay of the viral life cycle. We analyze the steady states and dynamical behavior of the model. We deduce a threshold parameter $\mathcal{R}_0$ (average number of newly infected cells produced by a single infected cell) in terms of the treatment efficacy parameter $0 \leq \varepsilon < 1$ and other parameters. The suggested model has the ability to provide accurate descriptions of nonlinear biological systems with memory. The obtained results give an insight to understand the dynamics of HCV infection.

Keywords: Mathematical modeling; Hepatitis C virus; Interferon-α; Viral dynamics; Stability

MSC2010: 34, 34C, 34A08, 34C60, 92, 92C, 92C42

1 Introduction

Hepatitis C (HCV) is an infectious disease which spreads through blood contact. It is estimated that HCV has infected about 200 million individuals worldwide [1]. About 50-80% of HCV infected cases are chronic in nature [2]. Of these chronic cases, about 10-20% develop into liver cirrhosis of which, about 5% develop hepatocellular carcinoma (HCC). The extend of prevalence of HCV varies widely across geographical locations. In most countries, the transmission of HCV occurs primarily through *injecting drug use* (IDU), which is mainly associated with the sharing of contaminated syringes/needles. Although evidence for risk of HCV infection through sharing of other injecting paraphernalia is increasing and giving rise to a menace. The absence of reliable screening for HCV amongst blood donors remains a major challenge in combating the spread of the disease in the eastern countries [3]. Geographically, HCV genotypes 1, 2, and 3 occur worldwide, whereas infection with HCV genotypes 4 and 5 occurs mainly in Africa, and HCV genotype 6 appears mainly in Asia [4]. Many mathematical epidemic models quantify the transmission of Hepatitis C among IDUs in the population. These models provide alternative means to define the problems, organize our the thoughts, understand the data, communicate, test understanding, and help in predictions among groups [5]. Now, there is a motivation for further study of the dynamics and mathematical modeling of hepatitis C virus in cellular level; See [6] and references therein.

Mathematical models, using ordinary differential equations with integer-order, have been proven valuable in understanding the dynamics of hepatitis C infections, in cellular level or in one host [7, 8]. Most of these models have been restricted to the short-term dynamics of the systems. One of the earliest models was proposed by Neumann *et al.* [9], who examine the dynamics of HCV in presence of Interferon-α (IFN-α) treatment. They find that the primary role of IFN is in blocking the production of virions from the infected hepatocytes. Dahari *et al.* [7] in a subsequent and improved model, take into account the homeostatic mechanisms for the liver by incorporating a growth function.

However, classical mathematical models with integer-orders ignore the intermediate cellular interactions and memory effects. For example, the

*Corresponding: frihan@uaeu.ac.ae (F.A. Rihan)

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

kinetic of the viral decline in patients responc to interferon-α is characterized by bi-phase sh following a delay about 8-9 hours, likely be the sum of interferon-a pharmacokinetics pharmacodynamics as well as the intracellular dc of the ciral life cycle [10]. Therefore, model of the biological systems by fractional-or differential equations has more advantages t classical integer-order mathematical modeling, which such effects are neglected [11, 12].

Fractional order differential equations are na rally related to systems with memory which ex in most of biological systems [13, 14]. Also, they closely related to fractals [15], which are abund in biological systems. Fractional derivatives embody essential features of cell rheological behavior and have enjoyed greatest success in the field of rheology [16]. This is due to the fact that fractional derivatives enable the description of the memory and hereditary properties inherent in various cells and processes. It has been deduced in [17] that the membranes of cells of biological organism have fractional order electrical conductance and then are classified in groups of non-integer order models. In this paper, we propose a system of *fractional-order differential equations* (FODEs) for modeling the dynamics of HCV to model as correctly as possible the dynamics of the target cell population: uninfected target cells, infected cells, and viral load in presence of antiviral interferon-α drugs (IFN). We assume here that the target cells of the model are hepatocytes.

2 Mathematical Model of HCV

The model that we shall consider for HCV infection is based on a three dimensional model given by Dahari *et al.* [7] and model of Neumann *et al.* [9]. They assumed a simplified view of HCV infection and describes the response to interferon therapy through the coupled evolution of three populations: the uninfected hepatocytes, the productively infected hepatocytes, and the free HCV virions with the following ordinary-differential equations

$$\begin{aligned} DH &= s - \mu_H H - k_1 V H, \\ DI &= k_1' V H - \mu_I I, \\ DV &= \mu_b I - \mu_V V. \end{aligned} \tag{1}$$

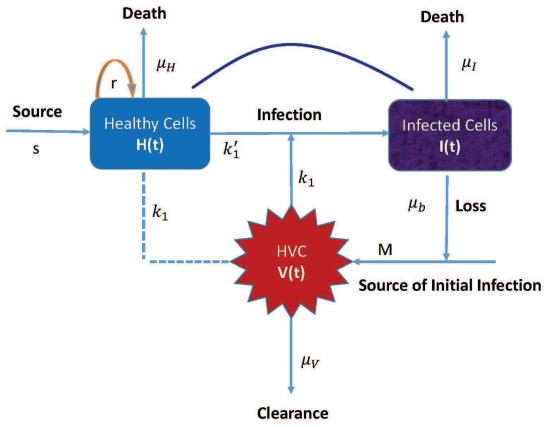


Figure 1: Schematic diagram showing the key-players in HCV infection models. $H(t)$ and $I(t)$ represent target and infected cells, respectively, and $V(t)$ represents free virus.

Here $D \equiv \dfrac{d}{dt}$, $H = H(t)$ represents the concentration of uninfected (healthy) hepatocytes, $I = I(t)$ is the concentration of infected hepatocytes, and $V = V(t)$ is the concentration of free HCV at time t. The model assumes that uninfected hepatocytes are produced at a constant rate s, die at rate μ_H per cell and are infected at constant rate k_1. Infected hepatocytes are lost at a rate μ_I per cell. The HCV, V, is assumed to infect the hepatocytes at a rate k_1, thereby producing infected hepatocytes, I, and k_1' is the rate at which infected cells become actively infected. Viral particles (virions) are produced at rate μ_b per infected hepatocytes and cleared at rate μ_V per virion.

We extend model (1) to include the logistic proliferation of uninfected hepatocytes and include fractional-order to the system to naturally relate the system with memory which exists in viral life cycle and time required for intracellular interactions. The fraction-order is successful in describing systems which have long-time memory and long-range interaction of the disease [18]. The modified model takes the form

$$\begin{aligned} D^{\alpha_1} H &= s - \mu_H H + rH\left(1 - \frac{H+I}{H_{max}}\right) - k_1 V H, \\ D^{\alpha_2} I &= k_1 V H - \mu_I I, \\ D^{\alpha_3} V &= (1-\varepsilon) M \mu_b I - k_1 V H - \mu_V V. \end{aligned} \tag{2}$$

for $0.5 < \alpha_i \leq 1$, $i = 1, 2, 3$. In model (2), we assume that the uninfected hepatocytes H are being produced at a rate s and proliferate

logistically $\left(1-\frac{H+I}{H_{max}}\right)$ at a rate r, accompanied by a natural death rate of μ_H, and H_{max} is the maximum hepatocyte count in the liver. We assume that the proliferation of infected cells is neglected, and physiological conditions $\mu_H H_{max} > s$, $r > \mu_H$. In the absence of any kind of treatment, the infected hepatocytes produce HCV at a rate μ_b, which has a clearance rate of μ_V. In this model, the impact of antiviral interferon-α drugs (IFN) on the dynamics of viral infection is considered by the coefficients $(1-\varepsilon)$, where ε is the efficacy of IFN. The viral production rate is then lowered by a fraction ε. μ_I is a blanket death term for infected cells, to reflect the assumption that we do not initially know whether the cells die naturally or by bursting. Since M viral particles are released by each lysing cell, this term is multiplied by the parameter M to represent the source for free virus (assuming a one-time initial infection); See Figure 1. The initial conditions for infection by free virus are $H(0) = H_0$, $I(0) = I_0$ and $V(0) = V_0$.

When hepatitis C virus first infects a person, the ensuing dynamics depend on the relative parameter values (see Table 1). Since newly infected individuals do not know that they are infected, we assume there is initially no treatment ($\varepsilon = 0$). We might expect several different scenarios: infection may fade out without becoming established, infection may spread with limited success and infect only part of the liver, or infection may spread rapidly and infect the whole liver. For untreated cronically infected with HCV, the mean serum viral load is approximately 3.5×10^6 IU/ml according to the WHO HCV RNA standard [19].

To understand the dynamics of system under acute infection correspond to each of these situations, it is helpful to walk through the stability of the steady states. The boundedness of the solutions is guaranteed by the following theorem.

Theorem 1 *The system (2) has a unique solution* $(H, I, V)^T$ *which remains in* $\mathbb{R}^3_+$ *and bounded by* H_{max}*; See [20].*

2.1 Equilibria and local stability of model (2)

To evaluate the equilibrium points of system (2), we put $D^{\alpha_1}H(t) = D^{\alpha_2}I(t) = D^{\alpha_3}V(t) = 0$. This model admits two steady states, namely the infection-free steady state $\mathcal{E}_0^* = (H_0, 0, 0)$, where

$$H_0 = \{r - \mu_H + [(r-\mu_H)^2 + 4rsH_{max}^{-1}]^{1/2}\}/2rH_{max}^{-1}, \quad (3)$$

and the infected state, $\mathcal{E}_+ = (H^*, I^*, V^*)$, where

$$\begin{aligned} H^* &= \frac{\mu_V \mu_I}{k_1' M(1-\varepsilon)\mu_b - k_1\mu_I}, \quad I^* = \frac{k_1' H^* V^*}{\mu_I}, \\ V^* &= \frac{\mu_I[(s+(r-\mu_H)H^*)H_{max} - rH^{*2}]}{H^*[k_1' rH^* - k_1\mu_I H_{max}]}. \end{aligned} \quad (4)$$

The Jacobian matrix $J(\mathcal{E}_0)$ for system (2) evaluated at the uninfected steady state $\mathcal{E}_0$ is then given by

$$J(\mathcal{E}_0) = \begin{pmatrix} -\mu_H + r - \frac{2rH_0}{H_{max}} & -\frac{rH_0}{H_{max}} & -k_1 H_0 \\ 0 & -\mu_I & k_1' H_0 \\ 0 & M(1-\varepsilon)\mu_b & -(k_1 H_0 + \mu_V) \end{pmatrix}. \quad (5)$$

Let us introduce the following definition and assumption to ease the analysis.

Definition 1 *The basic reproductive number of the virus* $\mathcal{R}_0$ *is defined as the average number of newly infected cells produced by a single infected cell at the beginning of the infection. The threshold parameter* $\mathcal{R}_0$ *has the property that if* $\mathcal{R}_0 < 1$*, then the endemic infected state does not exist, while if* $\mathcal{R}_0 > 1$ *the endemic infected state persists, where*

$$\mathcal{R}_0 = \frac{k_1'\mu_b(1-\varepsilon)MH_0}{\mu_I(\mu_V + k_1 H_0)}. \quad (6)$$

The uninfected steady state is asymptotically stable if all of the eigenvalues λ of the Jacobian matrix $J(\mathcal{E}_0)$, given by (5), have negative real parts. The characteristic equation $\det(J(\mathcal{E}_0) - I) = 0$ becomes

$$(\lambda + \mu_H - r + 2rH_0/H_{max})(\lambda^2 + B\lambda + C) = 0, \quad (7)$$

where $B = \mu_I + k_I H_0 + \mu_V$, and $C = \mu_I(k_1 H_0 + \mu_V) - k_1'\mu_b(1-\varepsilon)MH_0$. Hence, the three roots of the characteristic equation (7) are

$$\begin{aligned} \lambda_1 &= -\mu_H + r - rH_0/H_{max} \\ &\equiv -\sqrt{(r-\mu_H)^2 + 4rsH_{max}^{-1}} < 0, \quad (8) \\ \lambda_{2,3} &= \frac{1}{2}[-B \pm \sqrt{B^2 - 4C}]. \end{aligned}$$

Proposition 1 *If* $\mathcal{R}_0 \equiv \frac{k_1'\mu_b(1-\varepsilon)MH_0}{\mu_I(\mu_V + k_1 H_0)} < 1$*, then* $C > 0$ *and the three roots of the characteristic equation (7) will have negative real parts.*

Table 1: Parameter definitions and values used in the model.

P	Description	Units	Value	Source
s	Production rate of uninfected hepatocytes	cell ml^{-1}day^{-1}	0.1	[7]
μ_H	Natural death rate of rate of effector cells	day^{-1}	0.6	[7]
r	Proliferation rate of uninfected hepatocytes	day^{-1}	0.05	
H_{max}	maximum hepatocyte count in the liver	cells ml^{-1}	2×10^6	[7]
k_1	Infection rate of hepatocytes	ml day^{-1} virions^{-1}	0.08	[10]
k_1'	The rate o infected cells become infected	ml day^{-1} virions^{-1}	0.45	
μ_I	Natural death rate of infected cells	day^{-1}	0.28	[10]
μ_b	Production rate of HCV by the infected cells	cell $^{-1}$day^{-1}	1×10^{-4}	
μ_V	Clearance rate of HCV	virions virgins day^{-1}	2×10^{-2}	[7]
M	Source of free virus in the initial infection	virion	200	[10]
ε	The efficacy of IFN	–	$0 \leq \varepsilon < 1$	–

Assume that

$$\mathcal{R}_0^* = \frac{k_1' \mu_b M H_0}{\mu_I(\mu_V + k_1 H_0)} > 1 \geq \mathcal{R}_0. \tag{9}$$

Then, under the physiological conditions: $\mu_H H_{max} > s$, and $r > \mu_H$, we arrive at the following Remark [21].

Remark 1 *In case of uninfected steady state $\mathcal{E}_0$, we have three cases:*

(i) If $\mathcal{R}_0 < 1$, the uninfected state is asymptotically stable and the infected steady state $\mathcal{E}_+$ does not exist (unphysical). The efficacy of the drug ε should exceed $\left(1 - \frac{1}{\mathcal{R}_0^*}\right)$ to eradicate the virus.

(ii) If $\mathcal{R}_0 = 1$, then $C = 0$ and from (7) implies that one eigenvalue must be zero and the remaining two eigenvalues have negative real parts. The uninfected and infected steady state collide and there is a transcritical bifurcation, and the efficacy threshold is $\varepsilon^* = \left(1 - \frac{1}{\mathcal{R}_0^*}\right)$.

(iii) If $\mathcal{R}_0 > 1$, then $C < 0$, and thus at least one eigenvalue will be positive real root. Thus, the uninfected state $\mathcal{E}_0$ is unstable and the endemically infected state $\mathcal{E}_+$ emerges. The efficacy ε does not exceed $\left(1 - \frac{1}{\mathcal{R}_0^*}\right)$.

To study the local stability of the positive infected steady states $\mathcal{E}_+$ for $\mathcal{R}_0 > 1$, we consider the linearized system of (2) at $\mathcal{E}_+$. The Jacobian matrix at $\mathcal{E}_+$ becomes

$$J(\mathcal{E}_+) = \begin{pmatrix} -L^* & -rH^*/H_{max} & -k_1 H^* \\ k_1' V^* & -\mu_I & k_1' H^* \\ k_1 V^* & M\mu_b & -(k_1 H^* + \mu_V) \end{pmatrix}. \tag{10}$$

Here

$$L^* = -[\mu_H - r + k_1 V^* + r(2H^* + I^*)/H_{max}].$$

Then the characteristic equation of the linearized system is

$$P(\lambda) = \lambda^3 + a_1\lambda^2 + a_2\lambda + a_3 = 0, \tag{11}$$

$$\begin{aligned} a_1 &= \mu_I + \mu_V + k_1 H^* + L^*, \\ a_2 &= L^*(\mu_I + \mu_V + k_1 H^*) + \mu_I(\mu_V + k_1 H^*) - k_1^2 H^* V^* \\ &\quad - k_1' H^*(M\mu_b - rV^*/H_{max}), \\ a_3 &= k_1' H^*[k_1 M \mu_b V^* - r\mu_V V^*/H_{max} - L^* M \mu_b] + \\ &\quad L^* \mu_I(\mu_V + k_1 H^*) - \mu_I k_1^2 H^* V^*. \end{aligned}$$

The infected steady state $\mathcal{E}_+$ is asymptotically stable if all of the eigenvalues have negative real parts. This occurs if and only if Routh-Hurwitz conditions are satisfied, i.e. $a_1 > 0$, $a_3 > 0$ and $a_1 a_2 > a_3$.

3 Numerical Simulations

In this section, we carry out some numerical simulations to display the qualitative behaviours of model (2) (see Figures 2–3), and fit model (2)

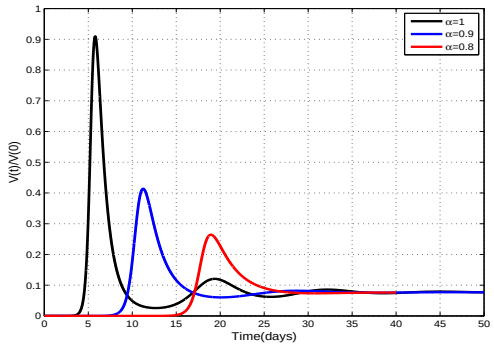


Figure 2: Numerical simulations of the HCV model (2) that show a stable infected steady state $\mathcal{E}_+$, with different fractional order α, with parameter values given in Table 1 and $\varepsilon = 0.0$ ($\mathcal{R}_0 > 1$).

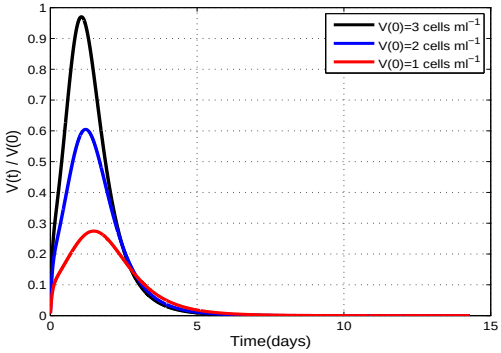


Figure 3: Simulations of the HCV model (2) which display a stable infection-free steady state $\mathcal{E}_0$, with the same parameter values of Figure 1 but with $r = 0.1$ (increasing proliferation rate of the uninfected hepatocytes) and $k_1 = 0.01$ (decreasing the infection rate of hypatocytes). A complete recovery is obtained when $\mathcal{R}_0 < 1$.

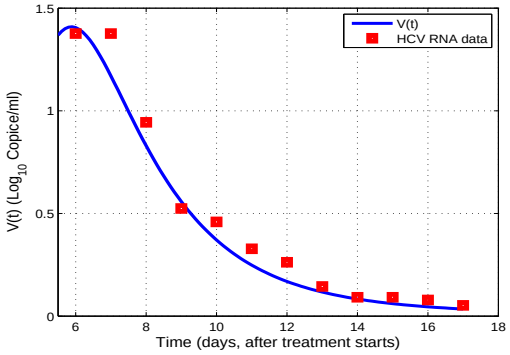


Figure 4: Comparison of viral load data (squares) with model predictions (2) for a case of HCV RNA decay profile during antiviral therapy, INF-α [22]. The treatment efficacy, as an estimate, $\varepsilon = 0.950$. Other parameter values and estimates are given in the text.

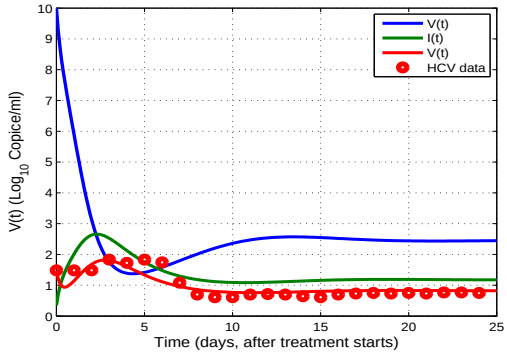


Figure 5: Comparison of viral load data with model predictions for a case of HCV RNA persist (endermic) steady state during INF-α therapy [22]. The treatment efficacy, as an estimate, $\varepsilon = 0.701$.

to experimental data of HCV RNA replications (see Figure 4– 5). The numerical simulations confirm the theoretical results obtained in the above sections. Figure 2 shows an asymptotically stable infected steady state $\mathcal{E}_+$ for different values of the fractional order and parameter values given in Table 1, when $\mathcal{R}_0 > 1$. While, Figure 3 shows a infection-free steady sate $\mathcal{E}_0$, when $\mathcal{R}_0 < 1$. A complete recovery is obtained when $\mathcal{R}_0 < 1$. Before treatment $\varepsilon = 0$, a steady state exists where viral production is balanced by viral clearance and the production of infected cells is balanced by their loss. Uninfected hepatocytes are also in steady state determined by the balance between their production, death, and loss due to infection. We notice that the smaller value of the fractional order α the longer incubational period of the virus in the beginning stage.

In Figure 4, we fit the model (2) to the experimental data of Table 2, during antiviral therapy ($0 < \varepsilon < 1$) for HCV infected patient. We fixed all the parameters except $P = [r, k_1, \mu_V, \varepsilon]$. The rest of the parameter take the values $s = 0.1 \times 10^2$, $k_1' = 0.0103$, $M = 800$, $H_{max} = 1.4 \times 10^3$, $\mu_H = 0.0107$, $\mu_I = 0.31$. Using least squares approach, the unknown parameters are $\hat{P} = [0.0401, 0.017, 0.731, 0.601]$. The reproductive number for the best estimate and infection-free steady state is $\mathcal{R}_0 = 0.7654 < 1$. The decay occurs rapidly during the treatment and the efficacy of treatment in blocking virion production $\varepsilon = 0.950$. The simulation match with the viral-free steady state $\mathcal{E}_0$. Figure 5 shows fitting the model (2) to the real data of Table 3 for chronically infected patient, during treatment. The parameter estimates with such data are $\hat{P} = [0.004, 0.021, 1.701, 0.502]$.

We employed the implicit Euler's scheme to solve the resulting biological (2). Interesting numerical simulations of the fractional-order model (2), with step-size $h = 0.05$ and $0.5 < \alpha \leq 1$ and parameters values given in the captions.

From the analysis and numerical approximation, we arrive at the following Remark.

Remark 2 *The presence of a fractional-order in the model can lead to a notable increase in the complexity of the observed behavior, as the solution is continuously depends on all the previous states. This confirms that the fractional-order plays the role of memory and heredity [23].*

4 Conclusions

In this paper, we developed a mathematical model for hepatitis C dynamics to describe the interactions between healthy liver cells H, infected liver I, and virus load V. While the model is overly simple in that it does not account for the immune response to HCV infection, but it provides a complex dynamics due to the fractional-order derivative that considers the longer term behavior of the HCV kinetics. The basic reproductive number of the virus, $\mathcal{R}_0$, has been deduced in understanding the persistence of viral infections. If $\mathcal{R}_0 < 1$, the level of virus load and infected cells will monotonically decrease and ultimately be eliminated. However for $\mathcal{R}_0 > 1$ there will be a chronic HCV infection. The higher the reproductive number $\mathcal{R}_0^*$, the higher treatment efficacy ε is required in order to eradicate the virus. When $\mathcal{R}_0 < 1$, the treatment efficacy ε greater than $(1 - 1/\mathcal{R}_0^*)$ leads to complete clearance of infection.

The model prediction is validated by fitting the model to available data for HCV RNA production for decay profile case and chronic infected case during treatment with interferon-α. When $\mathcal{R}_0 < 1$, the decay of the virion occurs rapidly during the treatment and the estimated efficacy of the drug (in blocking virion production) is $\varepsilon = 0.950$. While for the chronic state $\mathcal{R}_0 > 1$, the estimated efficacy parameter is $\varepsilon = 0.701$.

Acknowledgment

The work was funded by UAEU/SQU-2016 research projects (UAE University).

References

[1] WHO Fact Sheet 164-Hepatitis C, www.who.int.gate2.inist.fr/mediacentre/factsheets/fs164/en/, Januray 31, 2014.

[2] B. Roe, W. Hall, Cellular and molecular interactions in coinfection with hepatitis C virus and human immunodeficiency virus, Expert Rev Mol Med Oct 20 (2008).

Table 2: Hepatitides C verimemia within 14 days of treatment with interferon-α in a patient [9] (Case I).

Time (days)	1	2	3	4	5	6	7
$\log_{10} \bar{V}(t)$/liver	8.5	8.6	8.5	10.7	10.5	7.2	4.1
Time (days)	8	9	10	11	12	13	14
$\log_{10} \bar{V}(t)$/liver	3.5	2.6	1.6	0.7	0.6	0.4	0.3

Table 3: Hepatitides C verimemia within 25 days of treatment with interferon-α in a patient [9] (Case II).

Time (days)	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$\log_{10} \bar{V}(t)$/liver	9.6	10.2	8.5	10.1	10.2	8.2	6.2	4.5	3.6	3.6	4.0	3.8	4.2	4.1
Time (days)	15	16	17	18	19	20	21	22	23	24	25			
$\log_{10} \bar{V}(t)$/liver	3.7	3.5	4.0	4.2	4.3	4.2	4.3	4.2	4.4	4.4	4.3			

[3] A. Mukhopadhya, Hepatitis C in India, J. Biosci. 33 (4) (2008) 465–473.

[4] H. N. Afdhal, The natural history of hepatitis C, Semin Liver Dis 24 (Suppl 2:38) (2004) 5, 6.

[5] A. Wasley, M. J. Alter, Epidemiology of hepatitis C: geographic differences and temporal trends, Semin Liver Dis 20 (1) (2000) 116.

[6] I. Ramirez, Mathematical Modeling of Immune Responses to Hepatitis C Virus Infection, PhD thesis: East Tennessee State University, 2014.

[7] H. Dahari, A. Lo, R. M. Ribeiro, A. S. Perelson, Modeling hepatitis C virus dynamics: Liver regeneration and critical drug efficacy, J. Theor. Biol. 47 (2007) 371–381.

[8] A. S. Perelson, Modelling viral and immune system dynamics (2002).

[9] A. U. Neumann, N. P. Lam, H. Dahari, D. R. Gretch, T. E. Wiley, T. J. Layden, A. S. Perelson, Hepatitis C viral dynamics in vivo and the antiviral efficacy of interferon-α therapy, Science 282 (1998) 103–107.

[10] S. Zeuzem, E. Herrmann, Dynamics of hepatitis C virus infection, Ann Hepatol. 1 Apr-Jun (2) (2002) 56–63.

[11] F. A. Rihan, Numerical modeling of fractional-order biological systems, Abst. Appl. Anal. 2013 (2013) 11 pages.

[12] F. Rihan, S. Lakshmanan, A. Hashish, R. Rakkiyappan, E. Ahmed, Fractional order delayed predator-prey systems with holling type-ii functional response, Nonlinear Dynamics 80 (1) (2015) 777–789.

[13] F. A. Rihan, Current Topics in Salmonella and Salmonellosis (Ed.: Mihai Mares): Dynamics of Salmonella Infection, InTech, 2017.

[14] R. L. Magin, Fractional calculus models of complex dynamics in biological tissues, Comput Math Appl 59 (2010) 1586–1593.

[15] A. Rocco, B. J. West, Fractional calculus and the evolution of fractal phenomena, Physica A 265 (1999) 535.

[16] V. D. Djordjević, J. Jari, B. Fabry, J. J. Fredberg, D. Stamenovi, Fractional derivatives embody essential features of cell rheological behavior, Annal. Biomed. Eng. 31 (2003) 692–699.

[17] K. S. Cole, Electric conductance of biological systems, Cold Spring Harb Symp Quant Biol (1993) 107–116.

[18] E. Ahmed, H. A. El-Saka, On fractional order models for hepatitis C, Nonlinear Biomed Phys 4 (1) (2010) 1–4.

[19] J. M. Pawlotsky, M. Bouvier-Alias, C. Hezode, F. Darthuy, J. Remire, D. Dhumeaux, Standardization of hepatitis C virus RNA quantification, Hepatology 32 (3) (2000) 654–9.

[20] W. Lin, Global existence theory and chaos control of fractional differential equations, J. Math. Anal. Appl. 332 (2007) 709–726.

[21] F. A. Rihan, et al., Dynamics of hepatitis c virus infection: Mathematical modeling and parameter estimation, Math. Model. Nat. Phenom. 12 (5) (2017) 33–47.

[22] H. Dahari, et al., Mathematical modeling of primary hepatitis C infection: Noncytolytic clearance and early blockage of virion production, Gastroenterology 128 (2005) 1056–1066.

[23] F. A. Rihan, B. F. Rihan, Numerical modelling of biological systems with memory using delay differential equations, Appl. Math. & Inf. Sci. 9 (3) (2015) 1615–1658.

Identifying Translated uORFs based on Sequence Features via Tree-based Algorithms

Qiwen Hu[1] and Steffen Heber[2]
[1]Department of Systems Pharmacology and Translational Therapeutics, University of Pennsylvania
Philadelphia, PA, 19104, USA
[2]Department of Computer Science, North Carolina State University
Raleigh, NC, 27695, USA
qiwenhu@upenn.edu, sheber@ncsu.edu

Abstract

Upstream open reading frames (uORFs) are important post-transcriptional regulatory elements that occur in 5' untranslated region of mRNA molecules. Misregulation of translated uORFs has been associated with important diseases such as schizophrenia, bipolar affective disorder, and cancer. It is known that mRNA sequence features such as mRNA secondary structure and sequence context near the start codon are often associated with uORF translation, but details of the underlying mechanisms are still unclear. In this paper, we have used tree based learning algorithms in combination with ribosome profiling data generated from *Arabidopsis thaliana* to identify sequence features that are associated with translated uORFs. Using a boosting tree and the identified features set, we were able to predict 11962 translated uORFs, in 4152 genes. Our approach can be used even if ribosome profiling data is not available.

keywords: uORF translation, ribosome profiling, sequence features, tree based algorithm

1 Introduction

Upstream open reading frames (uORFs) are important post-transcriptional regulatory elements that occur in the 5' untranslated region (UTR) of mRNA molecules. Studies have shown that translated uORFs appear in a higher frequency in genes with important regulatory roles, such as growth factors and transcription factors [1], and that their misregulation may lead to important diseases, including schizophrenia, bipolar affective disorder, and cancer [2]. Although uORFs are prevalent in many organisms - about 50% of the genes in human and mouse, and about 40% of the genes in Arabidopsis contain uORFs [3, 4], only few of them are experimentally validated. The biological functions and translational status of most uORFs are still unclear.

In general, translated uORF sequences attenuate the translation of the downstream main open reading frame, but not all uORF sequences are translated [5]. It has been shown that the sequence context near the start codon may affect how ribosome recognize an uORF [6, 7], however the details of the underlying mechanism are still unclear. Several methods to identify translation initiation sites, as well as translated open reading frames, including translated uORFs, have been developed [8, 9]. In our previous study, we have used ribosome profiling (ribo-seq) data to identify a set of possible translated uORFs in *Arabidopsis thaliana* [10]. Unfortunately, ~30% of genes were not transcribed in our samples. As a consequence, ribo-seq data is not available for these genes, and the translational status of their uORFs is still missing.

In this paper, we use tree based learning algorithms to identify sequence features that are associated with translated uORFs. We show that these sequence features can be used to predict translated uORFs without the use of additional ribo-seq data. Using a boosting tree in combination with the identified feature sets, we predict 11962 translated uORFs. We have compared our results with previous studies: our predictions include 89% of the reported, experimentally verified uORFs, and 84% of the reported, conserved uORFs.

2 Methods

2.1 Sequence Preprocessing

We used ribosome footprints and the corresponding RNA-seq data from a study in the model plant *Arabidopsis thaliana* (NCBI accession number SRP056795) [11]. After preprocessing, sequencing reads were aligned with the *Arabidopsis* genome sequence (version TAIR10, http://www.arabidopsis.org/) using Tophat and default parameters [12]. Only reads with length in the range 25 – 40bp that mapped uniquely were considered for the following analysis.

We have extracted uORF sequences using the same method as described in [10]. Subsequently, we performed an exhaustive search for all possible uORFs that start with a start codon (ATG) in the 5' UTR region and end with a stop codon (TAG, TAA or TGA) in the same reading frame. Finally, we assigned the aligned ribo-seq reads to uORFs and transcript regions according to the TAIR10 gene annotation.

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

2.2 Ribosome Profiling Feature Generation

Ribosome profiling features were generated using the distribution of ribo-seq reads around uORF regions. Ribosomes have different moving patterns during translation initiation, elongation and termination. The moving patterns are reflected by the distribution of ribo-seq reads in the corresponding regions of a translated uORF. Therefore, a translated uORF should show characteristic ribo-seq read distribution patterns.

We extracted 11 ribosome profiling features for each uORF. Below is the description of each feature; more details can be found in [10].

1. Distance from uORF to the nearest ribo-seq read peak: *distance*.
2. Density ribo-seq reads in uORF region: *density*.
3. Maximum of ribo-seq read density in uORF region: *max_density*.
4. Minimum of ribo-seq read density in uORF region: *min_density*.
5. Ribo-seq read density in the region left of the uORF: *density_left*.
6. Ribo-seq read density in the region right of uORF: *density_right*.
7. Variance of the rib-seq read distribution in the uORF region: *var*.
8. uORFscore: *uorf_score*. A measure of read periodicity [13].
9. Ribo-seq read coverage of the first 3 codons in uORF region: *fn_cov*.
10. Proportion of ribo-seq reads that are in the same reading frame as the uORF: *uorf_cov*.
11. Ribo-seq read coverage in the uORF region: *cov*.

2.3 Feature Extraction

We generated an extensive list of features for each uORF region. The features can be divided into two groups: sequence related features and peptide related features.

Sequence related features include length of uORF (*length*), relative position of uORFs (*order*), reading frame of uORF with respect to the main ORF (*rf*), secondary structure of uORF measured by minimum free energy (*mfe*), length of 5' UTR, coding sequence, and 3' UTR, distance between uORF start codon and translation initiation site of the main ORF, distance between uORF stop codon and translation initiation site of the main ORF, an indicator variable to show if the uORF region overlaps with other open reading frames, and 15 additional features that describe the sequence context near the start codon.

Peptide related features focus on the protein product of the uORF. Peptide related features include amino acid composition of the uORF region, for example frequency of each amino acid as well as frequencies of the amino acid groups tiny, small, aliphatic, aromatic, nonpolar, polar, charged, basic and acidic (35 features in total), molecular weight (*mw*), isoelectric point (*pi*), hydrophobicity (*hb*), instability index of the uORF peptide (*ins*) [14], protein interaction index (*bp*) [15], and the codon adaptation index of the uORF region (*cai*) [16].

2.4 Identification of uORF Translation Pattern and Model Learning

To improve the reliability of our features and to reduce noise, we have discarded all uORF regions that overlap with other uORFs, resulting in a set of 3854 uORFs. We then performed *k*-means clustering based on ribosome profiling features on the remaining uORFs. To determine the number of groups (*k*) in our uORF set we used the average silhouette value [17].

Subsequently, we used tree-based learning to identify characteristic sequence features in the different uORF groups and to predict translated uORFs. Tree-based learning algorithms have been successfully used for various machine learning tasks, and they are known for their accurate and robust performance, ease of interpretation, and the ability of learning non-linear relationships, see [18] for a detailed description. In our study, we have compared the performance of decision trees, random forests and boosting trees. A decision tree is a classification algorithm in which internal tree nodes represent attribute tests, edges correspond to test outcomes, and leaf nodes correspond to classification results (in our case the translation status of the uORF). Starting at the root, each data point will traverse a path through the decision tree until it reaches a leaf node where a prediction is made [18]. Random forests and boosting trees are ensemble approaches based on decision trees. A random forest uses bootstrap resampling to grow multiple decision trees, and combines their results [19]; boosting trees compute a weighted mixture of decision trees [20].

We used 5-fold cross validation to measure the performance of the different algorithms. The entire dataset was divided into 2 parts: a training set and a testing set. For each algorithm, the model was tuned in a 5-fold cross validation experiment using training data only. The model that performed best was applied to the test set.

3 Result

3.1 Ribosome profiling Features Show Different Pattern among uORF Groups

We extracted a set of ribosome profiling features which capture the moving pattern of translating ribosomes for all non-overlapping uORFs, see section 2.2. Most of our features are uncorrelated, or they show only a small positive Pearson correlation value. Only one feature pair:

ribo-seq read coverage (*cov*) and ribo-seq read coverage of the first 3 codons (*fn*) has a Pearson correlation coefficient of 0.8. Subsequently, we performed *k*-means clustering based on Euclidean distance to identify different ribosome profiling feature pattern. Average silhouette value was used to determine the number of groups (*k*) in the data. We calculated the average silhouette value for different values of *k* and found a clear drop at *k*=7 (figure 1), suggesting *k*=6 as an appropriate choice.

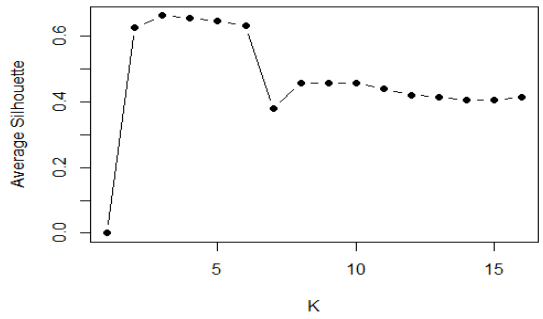


Figure 1: Average silhouette value of our *k*-means clustering for different values of *k*.

Figure 2 shows the different pattern of ribosome profiling features in the 6 groups. Similar to our results in [10], the ribosome profiling features show clear differences that suggest the translation status of the corresponding uORFs.

The uORFs in cluster 1 and 2 have very few ribosome profiling reads (the values of *density*, *density_left* and *density_right* are small) in their uORF regions, larger distance from the nearest density peak (*dp*) and a small variance of their read distribution (*var*). This suggests that ribosome profiling reads are neither accumulated at the start codon nor in the main body of the uORF. Likely, uORFs from these two groups are not translated.

The uORFs that belong to cluster 3, 5, or 6 have a smaller distance from the nearest density peak (*dp*) and an increased ribosome profiling read density (*denisty*) as compared to the uORFs in clusters 1 or 2. However, their read distribution has a small variance (*var*), which indicates that reads occur only at one or few positions with the uORF and do not show periodicity. This is inconsistent with the characteristics of translated open reading frames [9, 21].

Only the uORFs in cluster 4 show the characteristic features of translated uORFs [9, 21], such as accumulation of ribosome profiling reads at the start codon, high *uORFscore* and read periodicity, and an increased read density along the main body of the uORF (*min_density* > 0).

To further validate our results, we examined the distribution of experimentally verified and conserved uORFs that are expressed in our samples among the 6 groups (Figure 3). Consistent with ribosome profiling feature pattern, most of the experimentally verified and conserved uORFs are in cluster 4, only very few of them located in other clusters.

In summary, our analysis of ribosome profiling feature pattern, and the distribution of validated and conserved uORFs suggests that most of the translated uORFs are located in cluster 4.

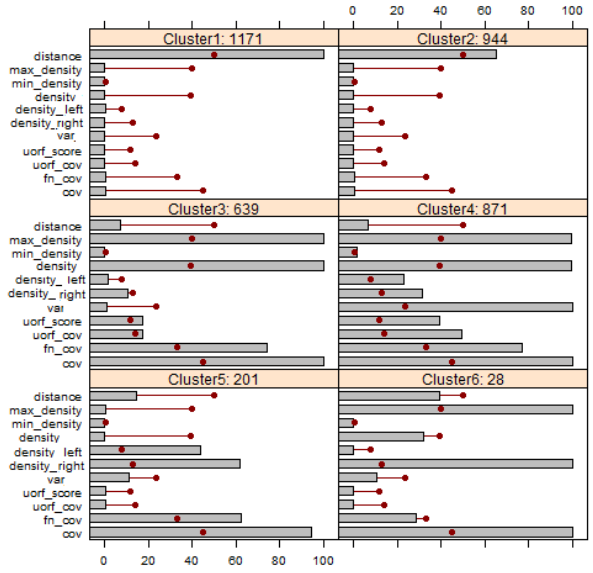


Figure 2: ribosome profiling features in different groups of our *k*-means clustering. The length of the bar indicates the range of each feature in the cluster. The red dot in the figure marks the mean value of a specific feature in the entire dataset.

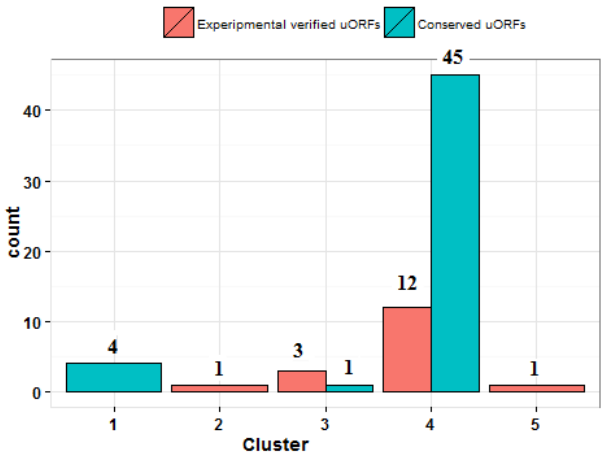


Figure 3: distribution of experimental verified and conserved uORFs among the different groups. Cluster 6 is not shown because there are no experimentally verified or conserved uORFs in this cluster.

Table 1: Model performance of our algorithms. RF: random forest, DT: decision tree, BT: boosting tree, RS: the ribo-seq feature based algorithm described in [26]. The number in the parentheses is the variance.

True data				
Algorithm	Accuracy	Precision	Recall	F1-score
RF	0.737 (6.4e-5)	0.759 (0.0018)	0.695 (0.0019)	0.725 (8.15e-5)
DT	0.686 (9.2e-4)	0.710 (0.0012)	0.648 (0.0098)	0.671 (5.3e-4)
BT	0.742 (3.2e-4)	0.764 (0.0019)	0.701 (4.9e-4)	0.730 (7.7e-5)
RS	0.9	0.95	0.87	0.91
Permutated data				
Algorithm	Accuracy	Precision	Recall	F1-score
RF	0.449 (9.5e-4)	0.457 (0.0034)	0.443 (0.0034)	0.440 (4.3e-4)
DT	0.486 (2.0e-4)	0.494 (7.3e-4)	0.386 (0.0017)	0.461 (7.7e-4)
BT	0.468 (6.4e-4)	0.475 (0.0026)	0.502 (0.0033)	0.482 (2.0e-4)

3.2 Sequence Features can be used to Predict Translated uORFs

Based on our clustering analysis, we classified the uORF sequences into two categories: translated uORFs (cluster 4) and untranslated uORFs (the other clusters). To obtain more reliable results, we also removed all uORFs with length smaller than 6bps; it has been reported that such small uORFs are less likely to be functional [22]. Finally, we extracted 59 sequence features from each uORF region (see methods).

To learn how well the extracted sequence features predict uORF translation, we applied three tree-based classification approaches (decision tree, random forest and boosting tree). We consider the sequence features as explanatory variables and the uORF translation status as class label. We have measured model performance in a 5-fold cross validation experiment using accuracy, precision, recall, and F1-score. The different quality measures are calculated as follows:

$$\text{Accuracy} = (TP+TN)/(TP+FN+TN+FP),$$

$$\text{Precision} = TP/(TP+FP),$$

$$\text{Recall} = TP/(TP+FN),$$

$$\text{F1-score} = 2TP/(2TP+FP+FN),$$

where TP, TN, FP, FN denote the number of true positives, true negatives, false positives, and false negatives.

Table 1 lists the performance of the three algorithms. Random forest and boosting tree show similar results and outperform a simple decision tree. To test the predictive power of each algorithm, we also permutated the response labels and recomputed the classifier performance; all results dropped to a value around 0.5. We also provide the performance of our previous, ribo-seq feature-based algorithm [23]. Since this algorithm uses ribo-seq information it performs better than our new algorithm. However, it cannot be used in cases where ribo-seq data is not available, for example in untranscribed genes.

3.3 Importance of Sequence Features for the Prediction of uORF Translation

To investigate how our sequence features affect the prediction of uORF translation, we calculated the importance score of each sequence feature. The importance score of a sequence feature is computed by subtracting the out of bag error derived from data where the sequence feature has been permuted from the out of bag error of the original data, and averaging over all trees [19].

Figure 4 shows the 20 features with the highest importance scores. The most important feature in our study is the start position of the uORF (*start*) in the 5'UTR. uORFs located at the beginning of the 5'UTR are more likely to be recognized by ribosomes and trigger translation. This is consistent with the findings of a previous study in human [24]. Our list includes several other features that have been reported in other studies about translation, for example uORF length (*length*), secondary structure near uORF region (measured by *mfe*), distance between uORF and its main open reading frame (*start_to_cds*) and codon usage (*cai*) in uORF region [6, 7, 25, 26]. Codon usage and secondary structure are two important factors that determine the translation efficiency of open reading frames during initiation and elongation stages [25, 26]. Additionally, our list includes the length of 3'UTR (*tutr*), molecular weight of the uORF peptide (*mw*), and length of main open reading frames (*cds*). Interestingly, we also found that amino acid composition (e.g. aromatic, aliphatic)

and frequency of specific amino acids (e.g. M and L) impact our predictions.

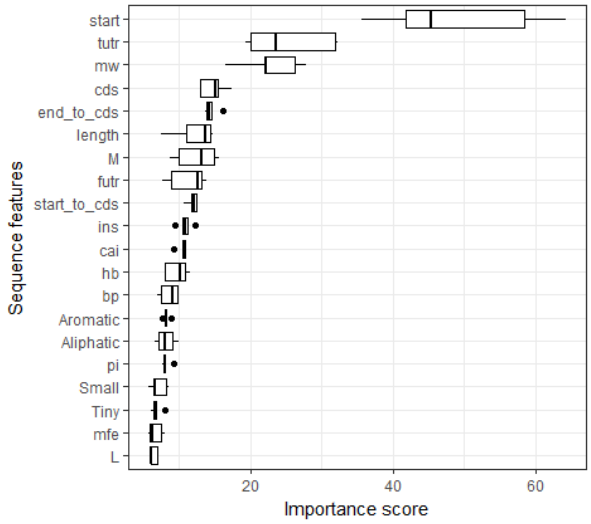


Figure 4: Top 20 sequence features with the highest importance score.

3.4 Genome-wide Identification of Translated uORFs

We used a boosting tree, the classifier that performed best in our evaluation, to predict translated uORFs in the genome of *Arabidospis thaliana*. We re-trained the model using the entire dataset. Model parameters were tuned in a 5-fold cross validation experiment via grid search. The best performance was achieved for following parameter set: iteration = 100, maximum tree depth *maxdepth* = 4, and boosting shrinkage parameter *nu* = 0.1.

Table 2: conserved and experimentally validated uORFs identified by our approach

	Predicted	Total	Recall
Experimentally verified	17	19	89.47%
Conserved	58	69	84.06%

Our approach predicted 11962 translated uORFs in 4152 genes using sequence features only. About 12% of the genes with predicted translated uORFs are untranscribed in our samples. Our approach predicts 1500 translated uORFs in these genes. A method which relies on ribo-seq information would not be able to detect these uORFs. We also compared our predictions with experimentally verified and conserved uORFs [3, 27-29]. Our approach detects 89% of the experimentally verified uORFs and 84% of the conserved uORFs (Table 2). Our previous ribo-seq feature-based algorithm detects considerably less uORFs (75% of the experimentally verified uORFs and 73% of the conserved uORFs) [10].

4 Conclusion

In this paper, we have identified sequence features that can be used to predict translated uORFs without the use of additional ribo-seq information. Using a boosting tree classifier in combination with the sequence features we predicted 11962 translated uORFs in the genome of *Arabidopsis thaliana*, promising candidates for future functional analyses. We have compared our results with previous studies: our predictions include 89% of the reported, experimentally verified uORFs, and 84% of the reported, conserved uORFs. Remarkably, our previous ribo-seq feature-based algorithm detects a considerably smaller percentage of these uORFs. We hypothesize that this difference is caused by genes for which ribo-seq information is not available (for example untranscribed genes), or difficult to interpret (for example genes with overlapping uORFs). However, not surprisingly, if sufficient ribosome profiling information is available, our previous ribo-seq feature-based algorithm performs better. We plan to combine both approaches in future work.

5 Acknowledgment

Parts of this work were supported by the National Science Foundation grant IOS1444561.

References

[1] Selpi, C. H. Bryant, G. J. Kemp, J. Sarv, E. Kristiansson, and P. Sunnerhagen, "Predicting functional upstream open reading frames in Saccharomyces cerevisiae," *BMC Bioinformatics*, vol. 10, p. 451, 2009.

[2] C. Barbosa, I. Peixeiro, and L. Romao, "Gene expression regulation by upstream open reading frames and human disease," *PLoS Genet*, vol. 9, p. e1003529, 2013.

[3] A. G. von Arnim, Q. Jia, and J. N. Vaughn, "Regulation of plant translation by upstream open reading frames," *Plant Sci*, vol. 214, pp. 1-12, Jan 2014.

[4] M. Matsui, N. Yachie, Y. Okada, R. Saito, and M. Tomita, "Bioinformatic analysis of post-transcriptional regulation by uORF in human and mouse," *FEBS Lett*, vol. 581, pp. 4184-8, Sep 04 2007.

[5] D. R. Morris and A. P. Geballe, "Upstream open reading frames as regulators of mRNA translation," *Mol Cell Biol*, vol. 20, pp. 8635-42, Dec 2000.

[6] A. G. Hinnebusch, "Molecular mechanism of scanning and start codon selection in eukaryotes," *Microbiol Mol Biol Rev,* vol. 75, pp. 434-67, first page of table of contents, Sep 2011.
[7] G. L. Chew, A. Pauli, and A. F. Schier, "Conservation of uORF repressiveness and sequence features in mouse, human and zebrafish," *Nat Commun,* vol. 7, p. 11663, May 24 2016.
[8] C. Fritsch, A. Herrmann, M. Nothnagel, K. Szafranski, K. Huse, F. Schumann, *et al.*, "Genome-wide search for novel human uORFs and N-terminal protein extensions using ribosomal footprinting," *Genome Res,* vol. 22, pp. 2208-18, Nov 2012.
[9] N. T. Ingolia, L. F. Lareau, and J. S. Weissman, "Ribosome profiling of mouse embryonic stem cells reveals the complexity and dynamics of mammalian proteomes," *Cell,* vol. 147, pp. 789-802, Nov 11 2011.
[10] Q. Hu, C. Merchante, A. N. Stepanova, J. M. Alonso, and S. Heber, "Genome-Wide Search for Translated Upstream Open Reading Frames in Arabidopsis Thaliana," *IEEE Trans Nanobioscience,* vol. 15, pp. 148-57, Mar 2016.
[11] C. Merchante, J. Brumos, J. Yun, Q. Hu, Kristina R. Spencer, P. Enríquez, *et al.*, "Gene-Specific Translation Regulation Mediated by the Hormone-Signaling Molecule EIN2," *Cell,* vol. 163, pp. 684-697.
[12] C. Trapnell, A. Roberts, L. Goff, G. Pertea, D. Kim, D. R. Kelley, *et al.*, "Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks," *Nat Protoc,* vol. 7, pp. 562-78, Mar 2012.
[13] A. A. Bazzini, T. G. Johnstone, R. Christiano, S. D. Mackowiak, B. Obermayer, E. S. Fleming, *et al.*, "Identification of small ORFs in vertebrates using ribosome footprinting and evolutionary conservation," *EMBO J,* vol. 33, pp. 981-93, May 02 2014.
[14] H. G. Boman, "Antibacterial peptides: basic facts and emerging concepts," *J Intern Med,* vol. 254, pp. 197-215, Sep 2003.
[15] K. Guruprasad, B. V. Reddy, and M. W. Pandit, "Correlation between stability of a protein and its dipeptide composition: a novel approach for predicting in vivo stability of a protein from its primary sequence," *Protein Eng,* vol. 4, pp. 155-61, Dec 1990.
[16] P. Rice, I. Longden, and A. Bleasby, "EMBOSS: the European Molecular Biology Open Software Suite," *Trends Genet,* vol. 16, pp. 276-7, Jun 2000.
[17] P. J. Rousseeuw, "Silhouettes: a Graphical Aid to the Interpretation and Validation of Cluster Analysis," *Computational and Applied Mathematics* pp. 53-65, 1987.
[18] Y. Y. Song and Y. Lu, "Decision tree methods: applications for classification and prediction," *Shanghai Arch Psychiatry,* vol. 27, pp. 130-5, Apr 25 2015.
[19] L. Breiman, "Random Forests," *Machine Learning,* vol. 45, pp. 5-32, 2001/10/01 2001.
[20] Z. Tu, "Probabilistic boosting-tree: learning discriminative models for classification, recognition, and clustering," *Tenth IEEE International Conference on Computer Vision (ICCV'05)* vol. 1, pp. 1589-1596 2005.
[21] N. T. Ingolia, "Genome-wide translational profiling by ribosome footprinting," *Methods Enzymol,* vol. 470, pp. 119-42, 2010.
[22] M. Cvijovic, D. Dalevi, E. Bilsland, G. J. Kemp, and P. Sunnerhagen, "Identification of putative regulatory upstream ORFs in the yeast genome using heuristics and evolutionary conservation," *BMC Bioinformatics,* vol. 8, p. 295, 2007.
[23] Q. Hu, C. Merchante, A. N. Stepanova, J. M. Alonso, and S. Heber, "Genome-Wide Search for Translated Upstream Open Reading Frames in Arabidopsis Thaliana," *IEEE Transactions on NanoBioscience,* vol. 15, pp. 148-157, 2016.
[24] S. E. Calvo, D. J. Pagliarini, and V. K. Mootha, "Upstream open reading frames cause widespread reduction of protein expression and are polymorphic among humans," *Proc Natl Acad Sci U S A,* vol. 106, pp. 7507-12, May 5 2009.
[25] M. dos Reis, R. Savva, and L. Wernisch, "Solving the riddle of codon usage preferences: a test for translational selection," *Nucleic Acids Res,* vol. 32, pp. 5036-44, 2004.
[26] Y. Lavner and D. Kotlar, "Codon bias as a factor in regulating expression via translation rate in the human genome," *Gene,* vol. 345, pp. 127-38, Jan 17 2005.
[27] A. Imai, Y. Hanzawa, M. Komura, K. T. Yamamoto, Y. Komeda, and T. Takahashi, "The dwarf phenotype of the Arabidopsis acl5 mutant is suppressed by a mutation in an upstream ORF of a bHLH gene," *Development,* vol. 133, pp. 3575-85, Sep 2006.
[28] F. Alatorre-Cobos, A. Cruz-Ramirez, C. A. Hayden, C. A. Perez-Torres, A. L. Chauvin, E. Ibarra-Laclette, *et al.*, "Translational regulation of Arabidopsis XIPOTL1 is modulated by phosphocholine levels via the phylogenetically conserved upstream open reading frame 30," *J Exp Bot,* vol. 63, pp. 5203-21, Sep 2012.
[29] A. Wiese, N. Elzinga, B. Wobbes, and S. Smeekens, "A conserved upstream open reading frame mediates sucrose-induced repression of translation," *Plant Cell,* vol. 16, pp. 1717-29, Jul 2004.

Scalable Approach to Data Driven Transcriptome Dynamics Modeling

Alexandr Koryachko, Samiul Haque, and Cranos Williams
Department of Electrical and Computer Engineering, NCSU
890 Oval Drive, 27606, Raleigh, USA
(akoryac, shaque2, cmwilli5)@ncsu.edu

Abstract

The evolving field of biological experimentation allows for the collection of various types of data describing different aspects of gene regulation inside a living cell. However, most of the gene expression dynamic modeling approaches limit their choice of data to a time course, which leads to infeasible requirements on the number of sampling time points to estimate the multitude of biologically relevant parameters. Thus, the model scope and parameter identifiability have to be sacrificed to approximate transciptome dynamics based on a typical number of time course samples. In this paper, we propose a scalable framework for building a model of transcriptome dynamics by aggregating a collection of experimental data available and suggest the types of additional experimentation to supplement the time course efficiently. The described approach is capable of increasing model descriptive and predictive power when additional data become available.

keywords: Gene Expression, Mathematical Modeling, Model Selection, Experimental Design, Nonlinear Dynamics, ODEs.

1 Introduction

Living organisms develop and respond to stimuli through a set of regulations on a molecular level. The regulation rules are hard-written in a genome and implemented through the action of transcription factors which modulate gene activity according to a given condition. Various types of experiments are performed to gain insight into that machinery to modify organisms in novel and strategic ways. Transcriptome abundance measurements are a widely utilized technique to estimate the change of gene activity over time or under a condition of interest. Methods of various complexity have been used to analyze the transcriptome (gene expression) data [24, 13]. Out of those methods the systems of Ordinary Differential Equations (ODEs) present the most descriptive way of representing transcriptome dynamics over time within a cell.

Ordinary Differential Equations (ODEs) gain a growing interest as a tool for modeling gene expression dynamics [24], yet a typical limitation of 2 to 8 time samples per time course [29] is still a barrier for a wide use in practical applications [2]. This limitation also leads to formulation of phenomenological models like linear models [4], Standardized Qualitative Dynamical Systems (SQUAD) models [20], or nonlinear basis functions models [8] with a small number of biologically irrelevant parameters rather than using mathematical constructs based on molecular kinetics like in S-Systems [23] or Hill-function kinetics based models [15]. Moreover, a wide range of proposed ODE structures for modeling transcriptome activity makes the choice of an appropriate mathematical representation challenging due to a lack of specific requirements for experimental data in the corresponding papers.

A number of studies have successfully applied ODEs to model transcriptome dynamics given a sufficient amount of information in terms of gene regulatory network graph and/or the results of various types of experiments which complemented the time course data [9, 4, 31]. Despite the findings facilitated by such modeling and the potential of building on previous results by collecting additional data, the cases of gradual model evolution are rather an exception than a rule. One such exception is the circadian clock effect in plants, which has been the subject of a number of ODE models [19, 18], continuously improved over time by the addition of new feedback loops [17], post-transcriptional and post-translational regulation [25], and mutant expression data [26]. In each case the addition of new data allowed for greater descriptive and predictive power [3]. However, each iteration required a reformulation of the previous model structure to incorporate new experimental results, making the process of model improvement long and not intuitive.

In this paper, we propose a methodology for dynamic model building which allows for a gradual increase in model complexity when new experimental data become available. In Section 2 we summarize the commonly used ODE structures into levels of mathematical complexity where each new level extends the

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

previous one based on additional data and propose the types of experiments allowing for an efficient transition between the levels. In Section 3 we propose criteria for data sufficiency at a given level of model complexity and an algorithm for aggregating the available experimental datasets. Thus, the resulting model will represent the outcomes of relevant experiments in a set of uniquely identifiable parameters, provide insights into transcriptome properties if the parameters are biologically relevant, and allow for gene expression predictions in a wide range of conditions combinations.

2 Model Formulation

2.1 Basic Model

Gene expression can be thought of as a balance between the rate of gene transcription and the rate of the corresponding mRNA degradation. Assuming both rates constant at a steady state, one can model gene expression dynamics with the following ODE:

$$\frac{dx}{dt} = a - bx, \tag{1}$$

where x represents gene expression, a represents the transcription rate ($a > 0$), and b represents the mRNA decay rate ($b > 0$). With a steady state assumption (i.e. $dx/dt = 0$) only one gene expression measurement x_{ss} would suffice to initiate the model building process and estimate the ratio of the rates at a steady state ($a/b = x_{ss}$).

Resolving between a and b requires additional experimentation. Time course data, the most common source of information in modeling approaches, can be used for this purpose if it captures a sufficient amount of gene expression dynamics. This scenario is rarely the case due to the typical sparseness of biological data. Zak et. al. proposed solving this problem by measuring the decay rate separately [35]. Barenco et. al. obtained direct mRNA decay rate measurements to constrain the tumor suppressor transcription factor p53 model while fitting it to the time course data [1]. Decay rate values may also be available in the literature [22, 32]. However, the reported values should be used with caution since decay rates are known to be condition specific [6]. Moreover, most experimental protocols are invasive and might heavily affect cellular physiology [21].

2.2 Transcription Factor Effect

Gene regulation in a cell is modulated through the activity of transcription factors. Assuming that transcription factors affect a common target gene x independently, this modulation can be reflected in Equation (1) as follows:

$$\frac{dx}{dt} = a f_1(x_1) f_2(x_2) \cdots f_R(x_R) - bx, \tag{2}$$

where a is a scaling coefficient, x_r ($r = 1, 2, \ldots, R$) is the expression of one of R transcription factors regulating x, and $f_r(x_r)$ is the regulator influence function which is equal to 1 when no regulation occurs, greater than 1 for activators, and between 0 and 1 for inhibitors. Influence function parameter estimation is heavily affected by the ability to differentiate between regulators' expression patterns based on sparse and noisy time course samples. Additional sampling time points or replicates do not guarantee sufficient resolution improvements. Thus, time course data should be supplemented with additional information to estimate the influence coefficients. Experiments where target expression is measured while regulator expression is manipulated can reveal this information.

Regulator knock-out mutant experiments [36, 14] can uniquely define a linear approximation of the influence function $f_r(x_r) = 1 + c_r x_r$. If transcription factor x_r is an activator with a measured wild-type expression x_r^{WT}, then target gene expression measurements in wild-type (x^{WT}) and mutant (x^{M_A}) conditions allow to approximate the regulator-target dependence with a line (Figure 1A):

$$x = x^{M_A} + \frac{x^{WT} - x^{M_A}}{x_r^{WT}} x_r,$$

which leads to a constant impact factor c_r^A approximation by associating x^{M_A} with the scaling coefficient a and rewriting the dependence in a form of the linear influence function:

$$f_{r_A}^{(lin)}(x_r) = 1 + \underbrace{\frac{x^{WT} - x^{M_A}}{x_r^{WT} x^{M_A}}}_{c_r^A} x_r.$$

However, a linear construct is expected to approximate the influence in a range that does not extend far beyond the regulator's wild type gene expression value. Otherwise, unrealistically high target expression is expected in case of activators and negative expression in case of inhibitors.

Hill-function approximation [7, 15] presents another, more biologically relevant, way of representing regulator influence (Figure 1B):

$$x = x^{M_A} + \left(x^{max} - x^{M_A}\right) \frac{x_r^l}{x_r^l + K^l} = x^{M_A} \cdot f_{r_A}^{(hill)}.$$

Here the target expression value under activator's influence is bounded. The bound estimate x^{max} can

be obtained through overexpression experiments [27] if the regulator's overexpression value is at least several fold larger than x_r^{WT}. The dissociation constant K can be estimated using knock-out mutant (x^{M_A}) and overexpression (x^{max}) experiment values. The regulator's protein affinity l can be obtained through additional experiments, for example, through fluorescence correlation spectroscopy in plants [5]. Hence, each additional parameter in the regulator influence function requires an experiment to estimate it.

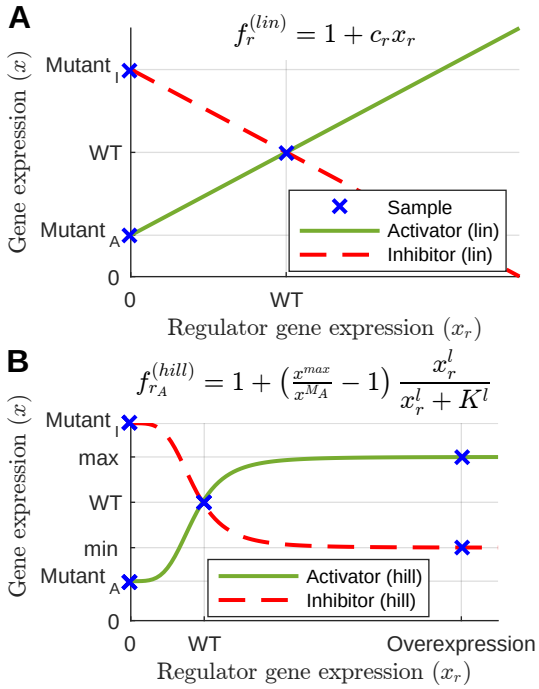


Figure 1: Influence function $f_r(x_r)$ under (A) Linear and (B) Hill-function approximation assumptions.

2.3 Condition Induced Effects

A host of transcriptome research studies are interested in mechanisms governing organism's response to a certain condition like biotic or abiotic stress in plants [12] or pathogen infection in single cells or animals [9, 34]. Equation (2) would be sufficient in describing gene expression dynamics over time under such condition if the full set of regulators is known, which is almost never the case at the current stage. Thus, the model has to account for unknown factors:

$$\frac{dx}{dt} = a f_u(t) \prod_{r=1}^{R} f_r(x_r) - bx, \tag{3}$$

where R is the number of known regulators, and $f_u(t)$ is a function aggregating the currently unknown influencing factors which change their activity under a condition of interest. An example of such influencing factor could be a change in a currently unknown condition induced transcription factor which binds to the target gene's promoter. $f_u(t)$ takes positive values and turns into 1 in wild-type conditions. The shape of $f_u(t)$ can be obtained using Gaussian process approximation [10]. Another approach would be to represent the unknown effect as a continuous shift to a new condition induced equilibrium:

$$u(t) = u_T \frac{1}{(\tau/t)^r + 1}, \tag{4}$$

where u_T represents an impact coefficient ($u_T > -1$), r quantifies how fast the transition between wild-type and condition induced steady states occurs ($r > 0$), and τ accounts for the transition delay (Figure 2). Because $u(t)$ turns to 0 when no condition is applied, an adjustment $f_u(t) = 1 + u(t)$ is needed to represent the unknown regulatory effect function. Parameters shaping $u(t)$ can be estimated by fitting the model to time course data under wild-type and the condition of interest.

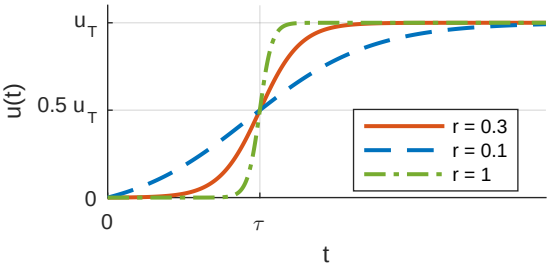


Figure 2: Sigmoid function approximation of unknown influencing factors effects. u_T – scale coefficient, r – rate coefficient , and τ – delay coefficient.

Additional experiments can help shaping the response to different levels of the applied condition if the condition levels are quantifiable and the sigmoid function is used. In such case the parameters shaping $u(t)$ are affected by the condition level S. We will concentrate on the condition dependence of the magnitude parameter u_T while the condition dependence of other two parameters from Equation (4) can be quantified in a similar fashion.

Wild-type and condition induced gene expression values allow for a linear approximation through a range of condition levels, which might, in some cases, be significantly far from reality. A more reliable approximation can be obtained by sampling gene expression at intermediate condition levels. However, transcriptome measurements are resource consuming, so the condition levels should be chosen in an efficient manner to produce maximum information with minimum experimentation. If the organism of interest exhibits a

quantifiable change in size, shape, or other easily accessible physiological parameters under the condition of interest, a faster and less expensive procedure of phenotyping can be used under a set of intermediate condition levels (e.g. micronutrient content level or pathogen load) to judge whether linear approximation captures the condition effect. Phenotyping results can also give clues on which condition levels to choose for the consequent transcriptome measurement experiments. Figure 3 shows a hypothetical example of selecting the most informative sampling point and the magnitude response function based on the results of phenotyping experiments.

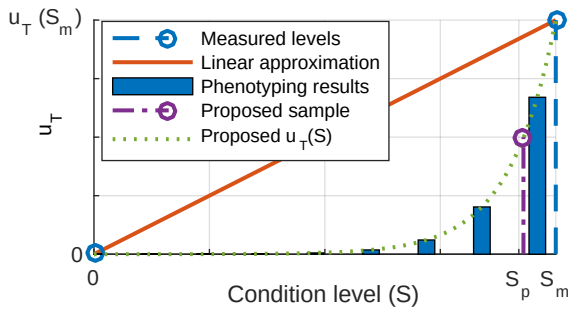


Figure 3: Condition level effect dependence modeling. S_m- measured condition level of the initial experiment, S_p-proposed condition level for gene expression measurements based on phenotyping experiments.

3 Model Fitting

Guidelines for the additional experimentation proposed in the previous section illustrate ways for increasing model complexity and, thus, its descriptive and predictive power. However, determining whether the collected data is sufficient for a given model complexity and combining various types of datasets to train the model are not trivial tasks considering that each type of experimentation has an associated measurement noise. We propose parameter identifiability analysis as a criteria for data sufficiency when scaling a model up and Bayesian inference for model parameter optimization.

3.1 Model Scalability Assessment

We require all parameters to be uniquely identifiable in the scaled up model representation to accept it. A parameter is considered non-identifiable if any deviation in its value produce an equally good model fit through the corresponding adjustments in other parameters. For example, any value of decay rate b can be compensated with a corresponding value of the transcription rate a in Equation (1) if x_{ss} is the only non wild type measurement at hand. Thus, parameter non-identifiability indicates a lack of data support for a given model structure. Several methods such as Differential Algebra Identifiability of Systems (DAISY) [30], Exact Arithmetic Rank (EAR) [11], and Profile Likelihood (PL) [28] have been used to detect non-identifiable parameters. Among these methods, PL is the only one that relies on experimental data in its identifiability analysis. We propose using the results of PL analysis for model discrimination when increasing model complexity based on additional data.

3.2 Parameter Estimation

Bayesian inference methods aggregate different sources of data by shaping prior distributions of the corresponding parameters before fitting a model to the corresponding time course. Each experiment that we proposed allows for obtaining mean and standard deviation estimates for a specific parameter. An assumption on experimental error distribution (e.g. Gaussian or Poisson) would allow to construct the corresponding prior distribution. The model fitting algorithm will sample parameter values from the corresponding prior distributions while minimizing the sum of squared differences between the time course data and gene expression pattern produced by the model. Parameter values from the regions that are far from the experimental measurements are highly unlikely to be sampled which would ensure that the model describes both the time course data and the results of the additional experiments.

Due to a nonlinear nature of the differential equations governing gene expression dynamics we suggest using the latest generation of Bayesian inference based parameter estimation algorithm, namely Differential Evolution Adaptive Metropolis (DREAM) software package [33]. It has been demonstrated that DREAM outperforms similar software in nonlinear, multimodal, and high dimensional problems [16].

4 Conclusion

In this paper, we presented a methodology for sequential increase in gene expression dynamic model complexity by aggregating different types of experimental data. The methodology provides a flexible framework for accumulating the existing knowledge of a biological process of interest at the transcriptome level and proposes efficient ways for expanding this knowledge through additional experimentation. This paper aims to facilitate modeling efforts in the studies where time course experiments have been implemented and the key regulatory connections have been identified.

References

[1] Martino Barenco, Daniela Tomescu, Daniel Brewer, Robin Callard, Jaroslav Stark, and Michael Hubank. Ranked prediction of p53 targets using hidden variable dynamic modeling. *Genome biology*, 7(3):R25, 2006.

[2] Daniel Brewer, Martino Barenco, Robin Callard, Michael Hubank, and Jaroslav Stark. Fitting ordinary differential equations to short time course data. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 366(1865):519–544, 2008.

[3] Nora Bujdoso and Seth J Davis. Mathematical modeling of an oscillating gene circuit to unravel the circadian clock network of arabidopsis thaliana. *Frontiers in Plant Science*, 4, 2013.

[4] Javier Carrera, Guillermo Rodrigo, Alfonso Jaramillo, Santiago F Elena, et al. Reverse-engineering the arabidopsis thaliana transcriptional network under changing environmental conditions. *Genome Biol*, 10(9):R96, 2009.

[5] Natalie M Clark, Elizabeth Hinde, Cara M Winter, Adam P Fisher, Giuseppe Crosti, Ikram Blilou, Enrico Gratton, Philip N Benfey, and Rosangela Sozzani. Tracking transcription factor mobility and interaction in arabidopsis roots with fluorescence correlation spectroscopy. *Elife*, 5:e14770, 2016.

[6] Nicole L Garneau, Jeffrey Wilusz, and Carol J Wilusz. The highways and byways of mrna decay. *Nature reviews Molecular cell biology*, 8(2):113–126, 2007.

[7] Rudolf Gesztelyi, Judit Zsuga, Adam Kemeny-Beke, Balazs Varga, Bela Juhasz, and Arpad Tosaki. The hill equation and the origin of quantitative pharmacology. *Archive for history of exact sciences*, pages 427–438, 2012.

[8] Mika Gustafsson, Michael Hörnquist, Jesper Lundström, Johan Björkegren, and Jesper Tegnér. Reverse engineering of gene networks with LASSO and nonlinear basis functions. *Annals of the New York Academy of Sciences*, 1158(1):265–275, 2009.

[9] Reinhard Guthke, Ulrich Möller, Martin Hoffmann, Frank Thies, and Susanne Töpfer. Dynamic network reconstruction from gene expression data applied to immune response during bacterial infection. *Bioinformatics*, 21(8):1626–1634, 2005.

[10] Ruirui Ji, Xinxin Zhang, and Xiaomei Yan. Modelling transcriptional regulation with fractional order differential equation using gaussian process. In *Control Conference (CCC), 2016 35th Chinese*, pages 9366–9370. IEEE, 2016.

[11] Johan Karlsson, Milena Anguelova, and Mats Jirstrand. An efficient method for structural identifiability analysis of large dynamic systems. *IFAC Proceedings Volumes*, 45(16):941–946, 2012.

[12] Joachim Kilian, Dion Whitehead, Jakub Horak, Dierk Wanke, Stefan Weinl, Oliver Batistic, Cecilia D'Angelo, Erich Bornberg-Bauer, Jörg Kudla, and Klaus Harter. The AtGenExpress global stress expression data set: protocols, evaluation and model data analysis of UV-B light, drought and cold stress responses. *The Plant Journal*, 50(2):347–363, 2007.

[13] Alexandr Koryachko, Anna Matthiadis, Joel J Ducoste, James Tuck, Terri A Long, and Cranos Williams. Computational approaches to identify regulators of plant stress response using high-throughput gene expression data. *Current Plant Biology*, 3:20–29, 2015.

[14] Alexandr Koryachko, Anna Matthiadis, Durreshahwar Muhammad, Jessica Foret, Siobhan M Brady, Joel J Ducoste, James Tuck, Terri A Long, and Cranos Williams. Clustering and differential alignment algorithm: Identification of early stage regulators in the arabidopsis thaliana iron deficiency response. *PloS one*, 10(8):e0136591, 2015.

[15] Jan Krumsiek, Sebastian Pölsterl, Dominik Wittmann, and Fabian Theis. Odefy-from discrete to continuous models. *BMC bioinformatics*, 11(1):233, 2010.

[16] Eric Laloy and Jasper A Vrugt. High-dimensional posterior exploration of hydrologic models using multiple-try dream (zs) and high-performance computing. *Water Resources Research*, 48(1), 2012.

[17] James CW Locke, László Kozma-Bognár, Peter D Gould, Balázs Fehér, Eva Kevei, Ferenc Nagy, Matthew S Turner, Anthony Hall, and Andrew J Millar. Experimental validation of a predicted feedback loop in the multi-oscillator clock of arabidopsis thaliana. *Molecular systems biology*, 2(1):59, 2006.

[18] James CW Locke, Megan M Southern, László Kozma-Bognár, Victoria Hibberd, Paul E Brown, Matthew S Turner, and Andrew J Millar.

Extension of a genetic network model by iterative experimentation and mathematical analysis. *Molecular systems biology*, 1(1), 2005.

[19] JCW Locke, AJ Millar, and MS Turner. Modelling genetic networks with noisy and varied experimental data: the circadian clock in arabidopsis thaliana. *Journal of theoretical biology*, 234(3):383–393, 2005.

[20] Luis Mendoza and Ioannis Xenarios. A method for the generation of standardized qualitative dynamical systems of regulatory networks. *Theoretical Biology and Medical Modelling*, 3:13, 2006.

[21] Sarah E Munchel, Ryan K Shultzaberger, Naoki Takizawa, and Karsten Weis. Dynamic profiling of mrna turnover reveals gene-specific and system-wide regulation of mrna decay. *Molecular biology of the cell*, 22(15):2787–2795, 2011.

[22] Reena Narsai, Katharine A Howell, A Harvey Millar, Nicholas O'Toole, Ian Small, and James Whelan. Genome-wide analysis of mrna decay rates and their determinants in arabidopsis thaliana. *The Plant Cell Online*, 19(11):3418–3436, 2007.

[23] Leon Palafox, Nasimul Noman, and Hitoshi Iba. Reverse engineering of gene regulatory networks using dissipative particle swarm optimization. *Evolutionary Computation, IEEE Transactions on*, 17(4):577–587, 2013.

[24] Chanda Panse, Dr Kshirsagar, et al. Survey on modelling methods applicable to gene regulatory network. *arXiv preprint arXiv:1310.2361*, 2013.

[25] Alexandra Pokhilko, Sarah K Hodge, Kevin Stratford, Kirsten Knox, Kieron D Edwards, Adrian W Thomson, Takeshi Mizuno, and Andrew J Millar. Data assimilation constrains new connections and components in a complex, eukaryotic circadian clock model. *Molecular Systems Biology*, 6(1), 2010.

[26] Alexandra Pokhilko, Paloma Mas, and Andrew J Millar. Modelling the widespread effects of toc1 signalling on the plant circadian clock and its outputs. *BMC systems biology*, 7(1):23, 2013.

[27] Gregory Prelich. Gene overexpression: uses, mechanisms, and interpretation. *Genetics*, 190(3):841–854, 2012.

[28] Andreas Raue, Clemens Kreutz, Thomas Maiwald, Julie Bachmann, Marcel Schilling, Ursula Klingmüller, and Jens Timmer. Structural and practical identifiability analysis of partially observed dynamical models by exploiting the profile likelihood. *Bioinformatics*, 25(15):1923–1929, 2009.

[29] Bruce A Rosa, Ji Zhang, Ian T Major, Wensheng Qin, and Jin Chen. Optimal timepoint sampling in high-throughput gene expression experiments. *Bioinformatics*, 28(21):2773–2781, 2012.

[30] Maria Pia Saccomani, Stefania Audoly, and Leontina D'Angiò. Parameter identifiability of nonlinear systems: the role of initial conditions. *Automatica*, 39(4):619–632, 2003.

[31] Yara-Elena Sanchez-Corrales, Elena R Alvarez-Buylla, and Luis Mendoza. The arabidopsis thaliana flower organ specification gene regulatory network determines a robust differentiation process. *Journal of theoretical biology*, 264(3):971–983, 2010.

[32] Kate Sidaway-Lee, Maria J Costa, David A Rand, Bärbel Finkenstadt, and Steven Penfield. Direct measurement of transcription rates reveals multiple mechanisms for configuration of the arabidopsis ambient temperature response. *Genome biology*, 15(3):R45, 2014.

[33] Jasper A Vrugt, CJF Ter Braak, CGH Diks, Bruce A Robinson, James M Hyman, and Dave Higdon. Accelerating markov chain monte carlo simulation by differential evolution with self-adaptive randomized subspace sampling. *International Journal of Nonlinear Sciences and Numerical Simulation*, 10(3):273–290, 2009.

[34] Shuang Wu, Zhi-Ping Liu, Xing Qiu, and Hulin Wu. Modeling genome-wide dynamic regulatory network in mouse lungs with influenza infection using high-dimensional ordinary differential equations. *PloS one*, 9(5):e95276, 2014.

[35] Daniel E Zak, Gregory E Gonye, James S Schwaber, and Francis J Doyle. Importance of input perturbations and stochastic gene expression in the reverse engineering of genetic regulatory networks: insights from an identifiability analysis of an in-silico network. *Genome research*, 13(11):2396–2405, 2003.

[36] Jie Zhang, Bing Liu, Mengshu Li, Dongru Feng, Honglei Jin, Peng Wang, Jun Liu, Feng Xiong, Jinfa Wang, and Hong-Bin Wang. The bhlh transcription factor bhlh104 interacts with iaa-leucine resistant3 and modulates iron homeostasis in arabidopsis. *The Plant Cell*, 27(3):787–805, 2015.

IsoRef Improves The Reference-Based Transcriptome Assembly Accuracy for RNA-Seq Data

Xiang Ao, Zicheng Zhao, and Shuaicheng Li
Department of Computer Science, City University of Hong Kong
Hong Kong, China
shuaicli@cityu.edu.hk

Abstract

Transcript reconstruction from mammal RNA-Seq data remains a challenging problem due to several biases, such as those from sequencing or mapping, the complexity of mammalian transcriptome generation from alternative splicing, fragmentary characteristics of reads, and from the unbalanced sequencing. Here, IsoRef, a reference-based transcriptome assembler for RNA-Seq data, is proposed. IsoRef investigates information from not only sequencing data, but from transcript annotation as well, in order to build accurate splice graphs. A flow balancing technique is proposed to reduce the impact of false positive transcripts and to narrow the search space of true positive transcripts. For each of two *in silico* datasets, IsoRef predicted 1,400 additional correct transcripts than StringTie; for each of the five actual datasets, IsoRef identified at least 1,500 additional correct transcripts than StringTie, which improves the transcript-level and gene-level accuracy compared to StringTie with a maximum improvement of 20%. IsoRef is available at `deepomics.org/module-instances/2CA682222F734424/`.

keywords: transcriptome assembly, splice graph, flow-balance graph.

1 Introduction

Transcriptome assembly from RNA-Seq data is a challenging and essential task in profiling, analyzing and understanding the transcriptome complexity and molecular mechanisms. It combines the upstream data preparation (such as sequencing library preparation and alignment of reads) and the downstream analysis (such as differential gene expression analysis and alternative splicing analysis) within the procedure of computational RNA-Seq data processing [3]. Due to the fragmentary characteristic of sequencing reads, the biases coming from various sources (such as sequencing biases and reads alignment biases, the complexity of mammalian transcriptome resulting from alternative splicing [15, 22]), the transcriptome assembly for mammals is full of difficulties and attracts a lot of interests from researchers [6, 14].

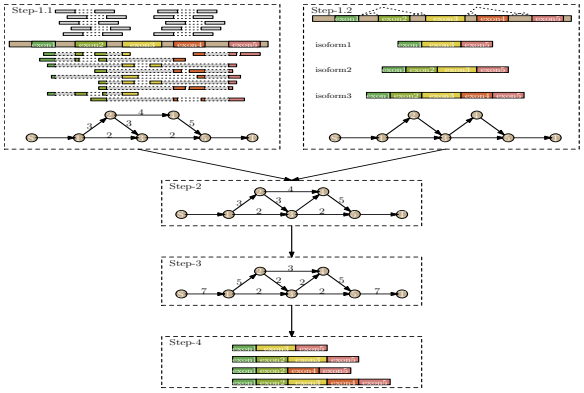


Figure 1: Schematic representation of IsoRef. In Step-1, IsoRef constructs two splice graphs from sequencing data and reference annotation, respectively. Then, IsoRef integrates these two graphs to produce a new comprehensive splice graph in Step-2. After balancing the flows of each node in the integrated graph in Step-3, IsoRef generates a flow-equilibrium splice graph and recovers transcripts in Step-4.

During the last decade, several transcriptome assembly methods were presented. Depending on the presence of the reference genome, these methods can be classified into two categories: the reference-based assembly methods (such as Cufflinks [21], IsoLasso [12], and StringTie [17]), and the *de-novo* assembly methods [6, 20] (such as Oases [18], ABySS [19], and Velvet [23]). The former methods require a reference genome for mapping of reads and potential transcripts are then derived from such alignments. Therefore, these methods are commonly used for organisms which genome has been well-annotated [2]. In contrast, the latter methods do not require reference genomes and deduct transcripts directly from sequencing reads. Such methods prevail in processing organisms without the existing genome [1]. Unsurprisingly, reference-guided assembly approaches usually produce transcripts with higher accuracy compared to the performance of *de novo* assemblers [13].

Among the reference-based assembly methods, there

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

are two popular assemblers gathering reputation from RNA-Seq data analyzers Cufflinks and StringTie. Cufflinks derives transcripts from an overlap graph in which nodes denote sequenced fragments and edges represent the connectivity between nodes supported by compatible fragments. Cufflinks adopts an ungenerous strategy to infer transcripts and hence produces a set of transcripts with the smallest values to construct the overlapped graph. As regards StringTie, it first builds a splice graph from aligned reads. The nodes in the splice graph, different from those in the overlap graph of Cufflinks, indicate continuous regions of the genome without splice interruption and edges correspond to the connectivity between nodes, in case there are supporting reads. Next, StringTie draws transcripts from the splice graph by applying a network flow algorithm. However, without using transcriptome annotation, both assemblers exhibited high error rates on real data [8].

Several factors can affect the performance of the existing assemblers. Firstly, there is a deficiency of sufficient reads to bridge the connections between the exons. Secondly, the reads can lead to inaccurate connections between exons, resulting in having false positives. Thirdly, the boundaries of exons are often wrongly defined, which leads to inaccuracy in the transcript inference. Aforementioned issues render assemblers difficult to reach a high accuracy level. Transcriptome annotations can bring improvement of the performance [8], since they provide the exon boundaries and their connectivity. In view of the fact that many mammalian reference transcriptomes are well-annotated, such annotations can be utilized in the assembly process.

Both Cufflinks and StringTie can employ the annotations, but using different strategies. Cufflinks produces the annotation transcripts as far as there are reads mapping to the transcripts, even when there is no alignment in specific exons of the transcripts. In contrast, StringTie generates the annotation transcripts only if the transcripts are feasible in the splice graph, which means that each component of the transcript has supporting reads. In general, Cufflinks produces transcripts according to the annotation and StringTie outputs transcripts in accordance with the reads alignments.

In this work, we propose IsoRef, a method emphasizing the importance of both the RNA-Seq data and the reference annotations. IsoRef takes the two types of data to reconstruct transcripts and estimates their respective expression levels simultaneously. Figure 1 displays the flowchart of IsoRef. Firstly, IsoRef constructs two splice graphs from sequencing data and the reference annotation (Step-1.1, Step-1.2), respectively. The two splice graphs are then integrated in order to generate a more comprehensive splice graph for subsequent procedures (Step-2). If the splice graph captures the actual expression levels of the exon, then a node in the graph (different from the source node and the sink node) should satisfy the flow balance property; that is, the amount of flows that arrived into the node is equal to the amount of flows that departed from it. Inspired by this insight, we proposed a flow-balance algorithm to balance flows, in order to obtain the flow-equilibrium splice graph (Step-3). Finally, this flow-equilibrium graph is adopted to recover transcripts (Step-4). Experimental results from the simulated data and the real data show the competitive performance of IsoRef, especially on the real data, in which IsoRef exceeded StringTie in terms of gene-level precision by 9.5% in average.

The rest of this paper is organized as follows: we present the IsoRef methods in detail in the methodology section. In Section 3 we compare the performance of IsoRef with Cufflinks and StringTie.

2 Methodology

IsoRef consists of four stages reconstructing transcripts from both RNA-seq data and the transcriptome annotation. At each gene locus, it firstly builds two splice graphs from sequencing data and annotations, respectively. Then, IsoRef combines the two splice graphs into a new splice graph. In step 3, IsoRef executes a flow-balance procedure on the comprehensive splice graph to get a flow-equilibrium splice graph, in which each node has the same amount of flows going in, through and out of the node. In step 4, it infers transcripts and calculates its corresponding abundance from the balanced splice graph.

Splice graph was first proposed in [9] and has been proven to perform well on many transcriptome assemblers, such as StringTie. For the sake of inferring transcripts precisely and fully using the transcript annotations, IsoRef constructs two splice graphs, if possible, for each gene pool. The first one is derived from reads and is called reads-derived splice graph (Step-1.1), and the other graph is derived from the relevant transcript annotations and is named as annotation-derived splice graph (Step-1.2). The nodes in reads-derived splice graph represent contiguous genomic regions where there is no interruption. Within the annotation-derived splice graph, these are exons or partial exons. Edges, on the other hand, indicate the connection relationships between nodes supported by either junction reads or annotation structures. A source node and a sink node are added to each graph, linking all starting nodes and ending nodes in the graphs, respectively. The two graphs may have different structures, and only reads-derived splice

graphs possess the flow in their nodes and edges. The amount of flow of each edge in the reads-derived splice graph can be calculated as the number of the supporting junction reads. On the other hand, the flows in nodes consider the node's average coverage as the corresponding value. Besides, flows of edges spreading out from the source node or into the sink node are recovered from the flows of connecting nodes.

In Step-2, IsoRef merges the splice graph derived from reads and the splice graph derived from annotation and creates a new more comprehensive splice graph. Nodes from the two graphs having the same definition and possessing identical connectivity with neighboring nodes are considered as concordant nodes. Those edges that link two identical nodes between the two graphs are concordant edges. The new splice graph retains all concordant nodes and edges and only manages those with inconsistency. Firstly, it excludes all nodes that do not possess any alignments. Nodes that have different definitions in the two graphs, but overlap in genomic coordinates are adjusted according to their definitions in annotation-derived splice graph, except those nodes which defined by junction reads. Edges are recovered from both the annotation-derived splice graph and the reads-derived splice graph as long as the connection between nodes exists. Loci having the reads-derived splice graph alone don't do this merging step, while those that have only the annotation-derived splice graph won't be delivered to subsequent processes.

Ideally, the flows entering in, going through and out of a specific node (except the source and the sink node) should be identical. IsoRef tries to achieve this state by applying a flow-balance algorithm in Step-3 (in this part we refer to IsoRef as IsoRef+). Suppose V and E are sets of nodes and edges of the integrated splice graph, respectively. Thus, $V = \{v_i | i = 0, 1, ..., N\}$, v_0 and v_N are the source node and the sink node, respectively. The set of edges is $E = \{\langle v_i, v_j \rangle | \forall i \in [0, N-1], \forall j \in [1, N], v_i, v_j \in V, \}$. , and f_{ij} indicates the flows in edge $\langle v_i, v_j \rangle$, where $f_i^{in} = \sum_{j=0}^{N-1} f_{ji}$ is the total number of flows entering to v_i and $f_i^{out} = \sum_{j=1}^{N} f_{ij}$ is the sum of flows going out of v_i, while f_i represents the number of flows going through node v_i. After balancing is performed, IsoRef is supposed to reach the state where $\forall i \in [1, N-1], f_i^{'in} = f_i^{'out} = f_i'$, in which $f_i^{'in}, f_i^{'out}, f_i'$ are the final three types of flows for the node i. IsoRef adopts the IBM CPLEX library and uses the linear programming solver to generate a new flow-balanced splice graph. Denote the flow change of the node i before and after balancing flow by ε_i and the change of edge $\langle v_i, v_j \rangle$ by ε_{ij}. Our objective of the linear programming problem is:

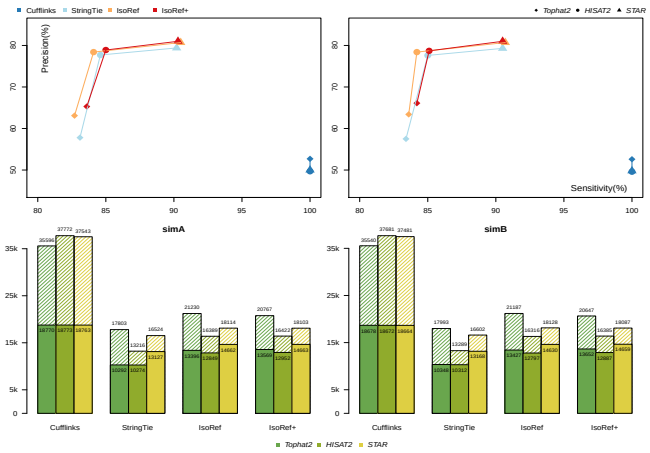


Figure 2: Performance of assemblers on simulated datasets (simA and simB). The upper two figures show the accuracy (precision and sensitivity) of assemblers. The bottom two figures display the total number of transcripts inferred by the assemblers, where the numbers of correctly inferred transcripts are colored.

$$min\left(\sum_{i=1}^{N-1}\varepsilon_i + \sum_{i=0}^{N-1}\sum_{j=1}^{N}\varepsilon_{ij}\right)$$
$$s.t. \quad \forall i \in [1, N-1], \quad \varepsilon_i \geq 0, \varepsilon_i^{in} \geq 0, \varepsilon_i^{out} \geq 0$$
$$-\varepsilon_i \leq f_i^{'} - f_i \leq \varepsilon_i, \qquad -\varepsilon_i^{in} \leq f_i^{in'} - f_i^{in} \leq \varepsilon_i^{in},$$
$$-\varepsilon_i^{out} \leq f_i^{out'} - f_i^{out} \leq \varepsilon_i^{out}, \qquad f_i^{'in} = f_i^{'} = f_i^{'out'};$$
$$\forall < v_i, v_j > \in E, \quad \varepsilon_{ij} \geq 0, \qquad -\varepsilon_{ij} \leq f_{ij}^{'} - f_{ij} \leq \varepsilon_{ij};$$

where f_{ij} *and* $f_{ij}^{'}$ are the flows in edge $\langle v_i, v_j \rangle$ before and after balancing, respectively.

Finally, IsoRef obtains transcripts from the flow-balance splice graph by using a similar method to the one proposed in StringTie. At each locus, IsoRef repeatedly searches the transcript (a path from the source to the sink node) from a node with the highest number of flows going through it. Then, it is extended in both directions to the source and to the sink node, by choosing the edges with the most abundant flows. IsoRef computes the magnitude of the flow in the transcript by applying the maximum flow algorithm. The process for inferring transcripts and its expression level repeats until the transcript's flow abundance is too low or there is no valid path available anymore.

3 Experimental Results

We compared IsoRef with two leading genome-based transcriptome assemblers, Cufflinks and StringTie, using both simulated and real data. All the assemblers were executed with default parameters. The reads data were aligned to hg19 genome reference using Tophat2 [11], HISAT2 [10] and STAR [4]. The performance of assemblers was evaluated in terms of accuracy consisting of precision and sensitivity. A predicted transcript is correct only if both boundaries of its introns match those of a reference transcripts

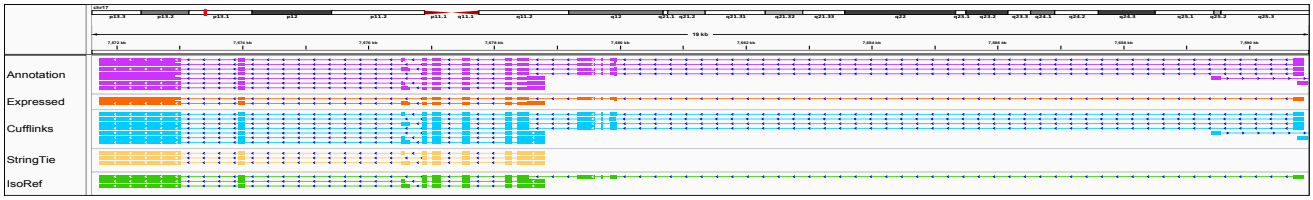


Figure 3: Comparison of predictions produced by Cufflinks, StringTie and IsoRef from HISAT2 alignments. Here, the predictions of gene TP53 made by the three assemblers are shown. The 'annotation' (purple) track shows the isoforms of TP53, the 'expressed' (orange) track shows the expressed isoforms, and 'Cufflinks' (cyan), 'StringTie' (yellow) and 'IsoRef' (green) tracks show the prediction results of corresponding assemblers.

and the number of exons is the same. The precision is the ratio of correctly derived transcripts and the number of predicted transcripts. The sensitivity is the ratio of correctly derived transcripts and the number of reference transcripts used in the evaluation. We used gffcompare [16] to evaluate the performance.

3.1 Performance on *in silico* data

We firstly compared IsoRef to other assemblers on simulated data, since the correctness of predicted transcripts can be precisely measured. We used Flux Simulator [7] to simulate RNA-Seq datasets. The parameters of Flux Simulator were set according to the RNA-Seq human protocol, which is provided on the simulator's website. Specifically, we simulated 150 million paired-end reads with 75bp length in each simulation. The reference genome was GRCh37/hg19 downloaded from UCSC Genome Browser. The corresponding transcriptome annotation of known RefSeq Genes was also downloaded from the browser

We simulated two datasets using the same protocol. However, as the Flux Simulator randomly selected transcripts to be expressed, at a stochastic expression level, such two datasets would be different. Therefore, we named them simA and simB, respectively. We adopted Tophat2, HISAT2 and STAR to align both datasets. The alignments were then passed to the assemblers. To assess the performance accurately, we employed the confidently expressed transcripts instead of hg19 annotation in the evaluation.

Figure 2 and Table 1 show the comparative results among the assemblers. Unsurprisingly, Cufflinks predominated in terms of sensitivity and the number of the correct transcripts, as it predicted transcripts once a reference annotation exists, regardless to its feasibility. However, the error rates of Cufflinks were also greatly above those of other assemblers (47.3%, 50.3% and 50% in simA, 47.4%, 50.5% and 50.2% in simB). Regarding the performance of StringTie and IsoRef, IsoRef predicted around 3100, 2400 and 1400 more correct transcripts from Tophat2, HISAT2 and STAR alignments, respectively, than StringTie, while the accuracy was not harmed. Besides, the expression correlations of IsoRef were improved compared to those of StringTie (Table 2). The benefits of using IsoRef, shown in the figures, demonstrate the superiority of IsoRef's strategy that incorporated information from annotation and reads. The IsoRef strategy helped discover more reliable transcripts, especially those that were not explicitly supported through reads. Its effectivity suggests the necessity of using the reference annotation in transcript reconstruction. Regarding IsoRef+, it predicted even more correct transcripts compared to IsoRef (above 150 more from Tophat2 alignments and around 100 more from HISAT2 alignments) and it had higher accuracy and expression correlations than both StringTie and IsoRef. The superiority of IsoRef+ compared to IsoRef indicates that balancing the flow of splice graph was beneficial to transcript reconstruction, increasing not only the accuracy of assembly, but also the number of correct transcripts. In Figure 3, we compared the performances of IsoRef, Cufflinks and StringTie in building the isoforms of gene TP53 from HISAT2 alignments. TP53, acting as a tumor suppressor, is the most studied gene in human genome [5]. From this figure, we know that TP53 has seven isoforms and only two of them were expressed during the simulation. Cufflinks predicted all seven isoforms, while StringTie gave only one correct prediction. In contrast, IsoRef built the structures for each of the expressed two isoforms.

3.2 Performance on real data

Next, the comparison on five real datasets is performed. We used IsoRef that undergoes the flow-balance technique. The real datasets are publicly available datasets, where three of them are from [17]. These datasets were downloaded from NCBI and are represented by "Lung" (GSM981244), "Blood" (GSM981256), "Monocytes" (GSM984609), "Spleen" (SRR4421334) and "HepG2" (SRR4422652) datasets. The dataset Spleen has 5.8 G bases of 100bp paired-end reads from a male adult's spleen, while the HepG2 is from the HepG2 cell line containing 2.1 G bases

of 50bp paired-end reads. Reads were mapped using Tophat2, HISAT2 and STAR as well.

Table 1: Accuracy of assemblers on simA and simB

		Cuff	ST	Ref	Ref+
TH+simA	sens	100	83.1	82.7	83.6
	prec	52.7	57.8	63.1	65.3
	called	35596	17803	21230	20767
	match	18770	10292	13396	13569
HT+simA	sens	100	84.6	84.1	85
	prec	49.7	77.7	78.4	78.9
	called	37772	13216	16389	16422
	match	18773	10274	12849	12952
SR+simA	sens	100	90.2	90.5	90.3
	prec	50	79.4	80.7	81
	called	37543	16524	18114	118103
	match	18763	13127	14662	14663
TH+simB	sens	100	83.4	83.6	84.2
	prec	52.6	57.5	63.4	66.1
	called	35540	17993	21187	20647
	match	18678	10348	13427	13652
HT+simB	sens	100	85	84.2	85.1
	prec	49.6	77.6	78.4	78.7
	called	37681	13289	16316	16385
	match	18672	10312	12797	12887
SR+simB	sens	100	90.5	90.7	90.5
	prec	49.8	79.3	80.7	81
	called	37481	16602	18128	18087
	match	18664	13168	14630	14659

'TH'=Tophat2, 'ST' = HISAT2, 'SR' = STAR, 'Cuff' = Cufflinks, 'ST' = StringTie, 'Ref' = IsoRef, 'Ref+' = IsoRef+, 'sens' = sensitivity, 'prec' = precision, 'called' = # of predicted transcripts, 'match' = # of correct predictions.

Table 2: Expression correlation on simulated data

		Cuff	ST	Ref	Ref+
TH+simA	pearson	0.9423	0.8337	0.8593	0.8753
	spearman	0.9435	0.8409	0.8479	0.8654
HT+simA	pearson	0.9381	0.8647	0.8756	0.8918
	spearman	0.919	0.8619	0.8709	0.8871
SR+simA	pearson	0.9578	0.8612	0.875	0.8925
	spearman	0.9576	0.85	0.8672	0.88
TH+simB	pearson	0.9449	0.8257	0.8378	0.871
	spearman	0.9448	0.8345	0.8368	0.8587
HT+simB	pearson	0.9413	0.8544	0.8663	0.8841
	spearman	0.9214	0.8544	0.864	0.8815
SR+simB	pearson	0.9562	0.8467	0.8692	0.886
	spearman	0.9565	0.8332	0.8626	0.874

Figure 4 and Table 3 show the performances of the assemblers on these datasets. Because in the actual data we cannot tell which transcripts were truly expressed, another criterion to evaluate the correct number of predicted genes is added. A predicted gene is considered correct if all the introns are exactly recognized in the order of an annotation. Thus, we can calculate the sensitivity and the precision for each assembler, at the gene level.

Similar to the results gained using *in silico* data, Cufflinks led the boards of sensitivity and precision at both transcript level and the gene level, as well as the board of the number of correct predictions. Since the exact transcripts that were expressed are unknown, the performance of Cufflinks was even better than its performance on simulated data. However, the number of false positive transcripts generated by Cufflinks is also higher when considering the above situation. Generally, for all datasets, IsoRef surpassed StringTie in most aspects of accuracy, no matter which tool used in alignment. On both Tophat2 and HISAT2 alignments, IsoRef held about 5% leads in major indices than StringTie, especially in the dataset HepG2, which displayed 22% and 21% higher precision at gene-level, respectively. Based on STAR alignments, IsoRef showed a slight advantage over StringTie at both transcript-level and gene-level accuracy. These results may be due to the higher rate of uniquely mapped reads using STAR than that using Tophat2 and HISAT2 (more than 7% unique mapping reads), which reduced the ambiguity of reads in the assembly and thus weakened the importance of annotation and the flow-balance algorithm IsoRef used. Regarding the number of correct transcripts, regardless of tools used in the alignment, IsoRef showed a stable improvement over StringTie on all datasets. Specifically, IsoRef produced about 2600, 1500 and 1600 more correct transcripts than StringTie on all the Tophat2, HISAT2 and STAR alignments, respectively.

4 Conclusion

We proposed a novel transcriptome assembly method IsoRef, which incorporates information from sequencing data and referred transcriptome annotation. The extra information from the annotation helps predict transcripts more precisely. Experimental results from simulated data and the real data both show the superiority of IsoRef over StringTie in transcripts prediction accuracy and expression estimation, regardless of the tools used in the alignment. Moreover, balancing the flow of the splice graph is proved to be a helpful technique to increase the number of correctly predicted transcripts, without harming the estimation of transcripts' expressions. .

References

[1] Caroline B Albertin, Oleg Simakov, et al. The octopus genome and the evolution of cephalopod neural and morphological novelties. *Nature*, 524(7564):220–224, 2015.

[2] Moran N Cabili, Cole Trapnell, et al. Integrative annotation of human large intergenic noncoding rnas reveals global properties and specific subclasses. *Genes & development*, 25(18):1915–1927, 2011.

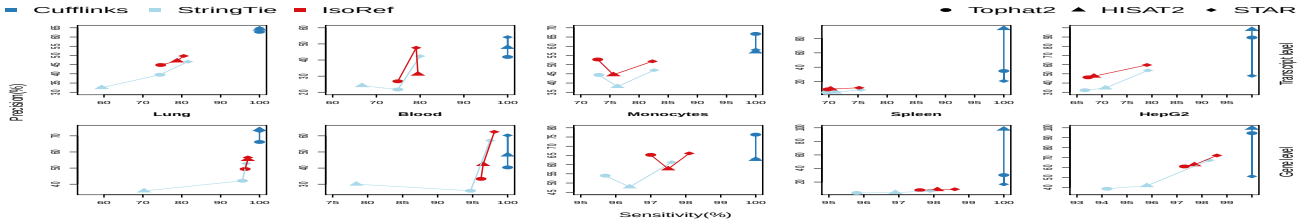
[3] Ana Conesa, Pedro Madrigal, et al. A survey of best practices for rna-seq data analysis. *Genome biology*, 17:13–13, 2016.

[4] Alexander Dobin, Carrie A Davis, et al. Star: ultrafast universal rna-seq aligner. *Bioinformatics*, 29(1):15–21, 2013.

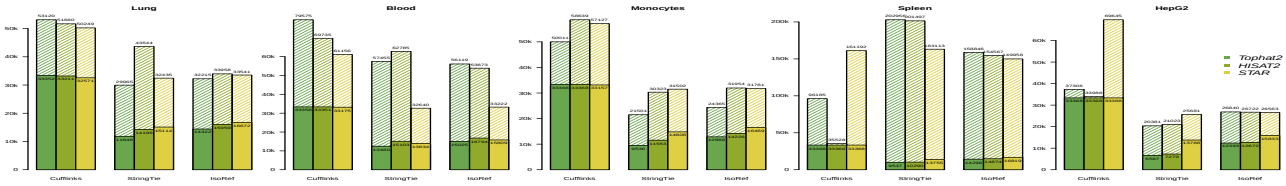
[5] Elie Dolgin. The most popular genes in the human genome. *Nature News*, 551:427–431, 2017.

[6] Liliana D Florea and Steven L Salzberg. Genome-guided transcriptome assembly in the age of next-generation sequencing. *IEEE/ACM transactions on computational biology and bioinformatics*, 10(5):1234–1240, 2013.

[7] Thasso Griebel, Benedikt Zacher, et al. Modelling and simulating generic rna-seq experiments with the flux



(a) Accuracy on real data



(b) Transcript count

Figure 4: Performance of assemblers on real datasets.

Table 3: Accuracy of assemblers on real datasets

	Lung				Blood				Monocytes				Spleen				HepG2			
	t-level		g-level		t-level		g-level		t-level		g-level		t-level		g-level		t-level		g-level	
	sens	prec	sens	prec	sens	prec	sens	prec	sens	prec	sens	prec	sens	prec	sens	prec	sens	prec	sens	prec
TH+Cuff	100	62.8	100	66.2	100	41.9	100	40.4	100	66.7	100	76.3	100	34.7	100	30.3	100	89.4	100	94.6
TH+ST	74.4	39.5	95.6	42.1	74.9	21.7	94.7	25.9	73.1	44.4	95.7	54	69.6	4.7	95.8	4	66.6	32.2	94.2	38.5
TH+Ref	74.6	44.8	96.3	49.4	74.9	26.8	96.2	33.3	72.9	52.8	97	65.3	69.8	9	97.6	8.1	67.2	46.2	97.3	61
HT+Cuff	100	64.3	100	73.9	100	47.8	100	48.1	100	56.9	100	62.8	100	93.9	100	98	100	98.2	100	99.5
HT+ST	59.4	32.5	70.3	35.8	66.7	24.1	78.4	30	76.3	38.1	96.4	47.9	71.1	5.1	96.9	4.4	70.7	34.6	95.8	41.6
HT+Ref	78.8	47	97	55.3	79.5	31.2	96.5	42.2	75.6	44.5	97.5	57.7	70.3	9.6	98.1	8.8	68.4	47.4	97.7	62.7
SR+Cuff	100	64.8	100	73	100	54.2	100	60.3	100	58	100	63	100	20.7	100	16.7	100	47.9	100	51
SR+ST	81.6	46.6	96.6	52.8	80	42.4	97.5	57.1	82.6	47	97.6	61.3	75.5	8.4	97.9	6.9	79.2	53.7	98.3	67.3
SR+Ref	80.5	49.7	97	56.6	79.1	47.6	98.1	62.5	82.3	51.8	98.1	66.1	75.2	11.2	98.6	9.4	79	59.6	98.6	72.1

'TH' = Tophat2, 'HT' = HISAT2, 'SR' = STAR, 'Cuff' = Cufflinks, 'ST' = StringTie, 'Ref' = IsoRef, 't-level' = transcript level, 'g-level' = gene level.

simulator. *Nucleic acids research*, 40(20):10073–10083, 2012.

[8] Katharina E Hayer, Angel Pizarro, et al. Benchmark analysis of algorithms for determining and quantifying full-length mrna splice forms from rna-seq data. *Bioinformatics*, 31(24):3938–3945, 2015.

[9] Steffen Heber, Max Alekseyev, et al. Splicing graphs and est assembly problem. *Bioinformatics*, 18(suppl 1):S181–S188, 2002.

[10] Daehwan Kim, Ben Langmead, and Steven L Salzberg. Hisat: a fast spliced aligner with low memory requirements. *Nature methods*, 12(4):357–360, 2015.

[11] Daehwan Kim, Geo Pertea, et al. Tophat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions. *Genome biology*, 14(4):R36, 2013.

[12] Wei Li, Jianxing Feng, and Tao Jiang. Isolasso: a lasso regression approach to rna-seq based transcriptome assembly. *Journal of Computational Biology*, 18(11):1693–1707, 2011.

[13] A Marchant, F Mougel, et al. Comparing de novo and reference-based transcriptome assembly strategies by applying them to the blood-sucking bug rhodnius prolixus. *Insect biochemistry and molecular biology*, 69:25–33, 2016.

[14] Jeffrey A Martin and Zhong Wang. Next-generation transcriptome assembly. *Nature Reviews Genetics*, 12(10):671–682, 2011.

[15] Qun Pan, Ofer Shai, et al. Deep surveying of alternative splicing complexity in the human transcriptome by high-throughput sequencing. *Nature genetics*, 40(12):1413–1415, 2008.

[16] Geo Pertea. gffcompare. `https://github.com/gpertea/gffcompare`.

[17] Mihaela Pertea, Geo M Pertea, et al. Stringtie enables improved reconstruction of a transcriptome from rna-seq reads. *Nature biotechnology*, 33(3):290–295, 2015.

[18] Marcel H Schulz, Daniel R Zerbino, et al. Oases: robust de novo rna-seq assembly across the dynamic range of expression levels. *Bioinformatics*, 28(8):1086–1092, 2012.

[19] Jared T Simpson, Kim Wong, et al. Abyss: a parallel assembler for short read sequence data. *Genome research*, 19(6):1117–1123, 2009.

[20] Tamara Steijger, Josep F Abril, et al. Assessment of transcript reconstruction methods for rna-seq. *Nature methods*, 10(12):1177–1184, 2013.

[21] Cole Trapnell, Brian A Williams, et al. Transcript assembly and quantification by rna-seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nature biotechnology*, 28(5):511–515, 2010.

[22] Eric T Wang, Rickard Sandberg, et al. Alternative isoform regulation in human tissue transcriptomes. *Nature*, 456(7221):470–476, 2008.

[23] Daniel R Zerbino and Ewan Birney. Velvet: algorithms for de novo short read assembly using de bruijn graphs. *Genome research*, 18(5):821–829, 2008.

Exploring Multi-Objective with Protein Sequence Alignment

Maha M. Abdelrasoul and Yaohang Li

Computer Science Department, Old Dominion University

Norfolk, VA, 23529, USA

{mabdelaa, yaohang}@cs.odu.edu

Abstract

The continuously growing protein sequence and structure data provide an increase in the need and importance of protein sequence alignment. Protein sequence alignment has been investigated by many researchers as a single-score optimization problem where both sequence and structural information are taken into account through a single combined score function. However, there is usually a trade-off between the sequence and structure scores, which makes it unlikely to generate a single alignment that optimizes all the scores. In this paper, we pursue a Multi-Objective Alignment (MOA) algorithm to obtain diversified alignments. The multi-objective alignment algorithm yields a better chance of obtaining the correct alignments and then achieving protein structure models with higher accuracy. The effectiveness of our multi-objective alignment algorithm is demonstrated in thread-based protein structure modeling on CASP11 targets.

1 Introduction

One of the essential tasks in bioinformatics is sequence alignment. As protein sequence alignment is fundamental to many problems in biology, such as protein structure modeling, protein design, and functional annotation of proteins [1] [2]. Generating an alignment between two protein sequences is generally done by optimizing an alignment scoring function. The most popular approach for protein sequence alignment is Dynamic Programming [3] [4] [5]. Several enhancements for the dynamic programming approach have been introduced. A remarkable one is the use of multiple sequence information to develop position-specific substitution profiles [6] [7] [8] [9]. Also the transformation of this multiple sequence alignment into a Hidden Markov Model [10] [11] [12] [13] is another addition to this technique. The position-specific substitution profiles have been used in several successful sequence alignment algorithms. One is CLUSTALW, which generates multiple sequence alignment in a pairwise manner [14] [15]. Another is SATCHMO, which simultaneously constructs a tree and a set of multiple sequence alignments, one for each node in the tree [16].

An important application of sequence alignment in protein structure modeling is thread-based protein structure modeling. Given a structure template, thread-based protein structure modeling aims at aligning every amino acid in the target sequence with the template sequence and evaluate how well the target fits into the template structure. Thread-based protein structure modeling is designed to model proteins that have the same fold as proteins with experiment-determined structures but without having high sequence similarity. In addition to sequence profile alignment, thread-based modeling also attempts to align structural information including secondary structures, solvent accessibility, backbone dihedral angles, and fragments [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] at the same time. The alignments of sequence and structural information are usually measured as individual scores, which are later linearly combined as a single alignment score. Dynamic programming or genetic algorithms are often employed to generate alignments by optimizing the alignment score [27] [28] [29] [30].

Correctly aligning the target sequence with the template sequence often yields maximum protein structure modeling accuracy for this template. Although typically the weights used to linearly combine individual scores can be trained by various machine learning algorithms, there is unlikely a single set of weights that can satisfy alignments of all target sequences and templates. This is due to the fact that the individual scores of sequence profiles and structural information are often conflicting. In this paper, we pursue a Multi-Objective Alignment (MOA) algorithm based on the Needleman-Wunsch algorithm [3] to obtain a set of diversified Pareto-optimal alignments. In theory, the multi-objective alignment algorithms can be considered as a super consensus method [31] [32] [33] whose goal is to derive all possible alignments with diversified consensus over all positive, linear weigh combinations. As a result, compared to finding a single alignment by optimizing a

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

certain combination of individual scores, the multi-objective alignment algorithm yields a better chance of obtaining the correct alignment and then achieving protein structure models with higher accuracy. The effectiveness of our multi-objective alignment algorithm is demonstrated in thread-based protein structure modeling on a set of CASP11 targets by deriving alignments based on sequence profile score and the structural information score.

2 Methodology

We examined the implementation of MOA algorithm for protein sequences based on the following objectives: (1) sequence profile, (2) secondary structure, and solvent accessibility objective functions.

Multi-Objective Alignment (MOA) Algorithm

Our idea for MOA is based on the Needleman-Wunsch algorithm, but instead of building only one score matrix we built a score matrix for each objective function. Tracing maximum-match pathway in each matrix will end up by generating the optimal alignment for the objective used to build this matrix. To get the multi-objectives alignments we will trace the maximum-match pathway in all the matrices to get each objective optimal alignment. Whenever these alignment decisions (match, insert, and delete) of the objectives disagree, a new alignment, which has the same starting part as the alignment being traced but continue by following the alignment decision of the disagreeing matrix, will be added. This procedure will be done until we generate all the alignments discovered while tracing the objective matrices. Finally, the scores of the generated alignments will be calculated according to all the objectives, and only the non-dominating alignments will be kept. The implementation of our method is split into two stages: score matrices generation and tracing objective matrices to generate the multi-objective alignments.

Score Matrices Generation

Given a set of objective functions $f_1(.), \dots, f_k(.)$, for two sequences $A = a_1 a_2 \dots a_M$ and $B = b_1 b_2 \dots b_N$, a score $s_{m,n}(f_i)$ is assigned to an aligned pair of residues a_m and b_n based on objective function $f_i(.)$. Besides, a gap penalty $g(f_i)$ is for aligning a residue from A/B to a gap. For each objective function $f_i(.)$, a score matrix $F(f_i)$ is computed according to Needleman-Wunsch algorithm and based on $f_i(.)$ scores, where $F_{m,n}(f_i)$ is calculated as follow:

$$F_{m,n}(f_i) = max \begin{cases} F_{m-1,n-1}(f_i) + s_{m,n}(f_i) & match/mismatch \\ F_{m-1,n}(f_i) + g(f_i) & insert \\ F_{m,n-1}(f_i) + g(f_i) & delete \end{cases}$$

The cells in $F(f_i)$ are generated one row at a time and one cell at a time starting from one at the up left corner. Once all the objective matrices are generated ($F(f_1), \cdots, F(f_k)$), the multi-objective alignments of sequences A and B with respect to $f_1(.), \dots, f_k(.)$. can be generated by tracing these matrices.

Backtracking the objective Matrices

Once the score matrices ($F(f_1), \cdots, F(f_k)$) are completely generated, the multi-objective alignments will be generated by backtracking. The difference here is that the backtracking is done in more than one matrix. The backtracking of the multi-objective alignments is performed using the following iterating steps:

1. Initialize a set of alignments U where U initially holds only one alignment U_1. An alignment U_j is represented by two empty strings $A_A \leftarrow$ "" and $A_B \leftarrow$ "" to hold store the alignment, and two indices $m = M$ and $n = N$ to keep track of the current index in each sequence.
2. For each alignment $U_j \in U$, trace the score at the cell of indices m, n in every score matrix $(F(f_1), \cdots, F(f_k))$, to determine the source of $F_{m,n}(f_1) \cdots F_{m,n}(f_k)$.
   a. If all $F_{m,n}(f_1) \cdots F_{m,n}(f_k)$ come from a match. Update U_j accordingly as $A_A \leftarrow a_m + A_A$, $B_B \leftarrow b_n + B_B$, $m = m - 1$, and $n = n - 1$.
   b. If all $F_{m,n}(f_1) \cdots F_{m,n}(f_k)$ come from an insert. Update U_j accordingly as $A_A \leftarrow a_m + A_A$, $B_B \leftarrow$ " $-$ " $+ B_B$, $m = m - 1$, and $n = n$.
   c. If all $F_{m,n}(f_1) \cdots F_{m,n}(f_k)$ come from a delete. Update U_j accordingly as $A_A \leftarrow$ " $-$ " $+ A_A$, $B_B \leftarrow b_n + B_B$, $m = m$, and $n = n - 1$.
   d. If $\exists\, F_{mn}(f_i)$ coming from an insert while others $F_{m,n}(.)$ come from match, a new alignment U_x is added based on the insert where $A_A \leftarrow a_m + A_A$, $B_B \leftarrow$ " $-$ " $+ B_B$, $m = m - 1$, and $n = n$. Also, U_j is updated according to the match as $A_A \leftarrow a_m + A_A$, $B_B \leftarrow b_n + B_B$, $m = m - 1$, and $n = n - 1$.
   e. If $\exists\, F_{mn}(f_i)$ that comes from a delete while the other $F_{m,n}(.)$ come from match, a new alignment U_x is added based on the delete where $A_A \leftarrow$ " $-$ " $+ A_A$, $B_B \leftarrow b_n + B_B$, $m = m$, and $n = n - 1$. Also, U_j is updated according to the match as $A_A \leftarrow a_m + A_A$, $B_B \leftarrow b_n + B_B$, $m = m - 1$, and $n = n - 1$.
   f. If $\exists\, F_{mn}(f_i)$ that comes from an insert while the other $F_{m,n}(.)$ come from delete, a new alignment U_x based on the insert where $A_A \leftarrow a_m + A_A$, $B_B \leftarrow$ " $-$ " $+ B_B$, $m = m - 1$, and $n = n$. Also, update U_j according to the delete as $A_A \leftarrow$ " $-$ " $+ A_A$, $B_B \leftarrow b_n + B_B$, $m = m$, and $n = n - 1$.
   g. If $\exists\, F_{mn}(f_i)$ that comes from an insert and $\exists\, F_{mn}(f_l)$ that comes from a delete while the other $F_{m,n}(.)$ come

from match, a new alignment U_x is added based on the insert where $A_A \leftarrow a_m + A_A$, $B_B \leftarrow " - " + B_B$, $m = m - 1$, and $n = n$. Also, a new alignment U_y is added based on the delete where $A_A \leftarrow " - " + A_A$, $B_B \leftarrow b_n + B_B$, $m = m$, and $n = n - 1$. Besides, U_j is added according to the match as $A_A \leftarrow a_m + A_A$, $B_B \leftarrow b_n + B_B$, $m = m - 1$, and $n = n - 1$.

3. Repeat step 2 until all the alignments in U reach indices 0,0.
4. For each alignment $U_j \in U$, calculate its score according to all the objectives.
5. Remove the dominated alignments from U.

Example

To demonstrate how the algorithm works, a simple alignment example is done over the following sequences.

Sequence A	P	Q	Q	Y	Y	P	Q	
Secondary Structure	C	H	H	B	B	C	C	
Sequence B	P	N	N	Y	Q	P	Y	Q
Secondary Structure	H	C	C	C	H	H	B	B

Where the objectives here are the profile and the secondary structure and the scoring function for both will be 1 for a match and -1 for mismatch or gap. Figure 1 shows an illustration of the alignments generation. Figure 2 show the scores for all the generated alignments, while Table 1 shows the resulting non- dominated ones.

3 Results

The Critical Assessment of Protein Structure Prediction (CASP) 11 targets are used to show the effectiveness of MOA. Here, we used two scoring functions to measure the alignment between the *i*th residue in the query sequence and the *j*th residue in the template sequence, which result in score matrices of the query and template sequences. The first one is based on the sequence profile such that

$$S_{seq}(i,j) = \sum_{k=1}^{20} Fa_q(i,k) + Fb_q(i,k)L_t(j,k)/2.$$

Here, $Fa_q(i,k)$ is the frequency of the kth amino acid at the ith position of the multiple sequence alignments (MSA) obtained by PSI-BLAST [34] against the non-redundant (NR) sequence database with an E-value cutoff of 0.001. $Fa_q(i,k)$ is considered as a close alignment frequency. $Fb_q(i,k)$ is a more distant frequency generated using a higher E-value cutoff of 1.0. The idea of combining distant and close sequence profiles comes from [28] [35] [36] [37], which helps increase the alignment sensitivity in different homologue areas. $L_t(j,k)$ is the derived log-odds profile of template sequence for the kth amino acid at the jth position. The template sequence derived log-odds profile generated from PSI-BLAST search with an E-value cutoff 0.001.

The second scoring function is based on structural features including predicted secondary structures and solvent accessibility.

$$S_{structure}(i,j) = SS(i,j) + SA(i,j).$$

Here, $SS(i,j)$ is the probability that the predicted secondary structure of the *i*th residue of the query sequence matches with that of the *j*th residue of the template sequence, i.e.,

$$SS(i,j) = Prob\left(ss_q(i) = ss_t(j)\right)$$

where $ss_q(i)$ and $ss_t(j)$ are the secondary structures of the *i*th residue of the query sequence and the *j*th residue of the template sequence. The secondary structure of the query is predicted by Scorpion [38] . Similarly, $SA(i,j)$ is the probability that the predicted solvent accessibility of the *i*th residue of the query sequence matches with that of the *j*th residue of the template sequence such that

$$SA(i,j) = Prob\left(sa_q(i) = sa_t(j)\right)$$

where $sa_q(i)$ and $sa_t(j)$ are the solvent accessibility of the *i*th residue of the query sequence and the *j*th residue of the template sequence. The solvent accessibility of the query is predicted by CASA [39].

MOA is compared with two popularly used template alignment and selection methods for template-based protein structure modeling (Muster [28] and GenTHREADER [40]). Each target sequence is aligned with the same templates by the structure profile alignment method. Then, tertiary protein structure models are generated by the Modeller program [41] according to the alignments. The Global Distance Test-Total Score (GDT-TS), which indicates the percentage of the model conformation superimposed correctly onto the native structure, is used to measure the quality of these models and the corresponding alignments. Since MOA generates more than one alignment, we only show the one with the highest GDT-TS score.

We first compare MOA and Muster on the top-ranked templates of each target specified by Muster. Figure 3 shows the GDT-TS score for Muster along with the MOA. As it appears in the figure that MOA achieved a higher or equal GDT-TS score for 102 targets and most of the time MOA is higher than Muster. Also MOA GDT-TS score was greater than Muster by at least 10 points in seven targets. Similar comparison is done between MOA and pGenTHREADER, which is shown in Figure 4. In 79 targets, the GDT-TS scores of models generated by MOA are higher than pGenTHREADER, where in 13 of them, the gain is at least 10 points or higher.

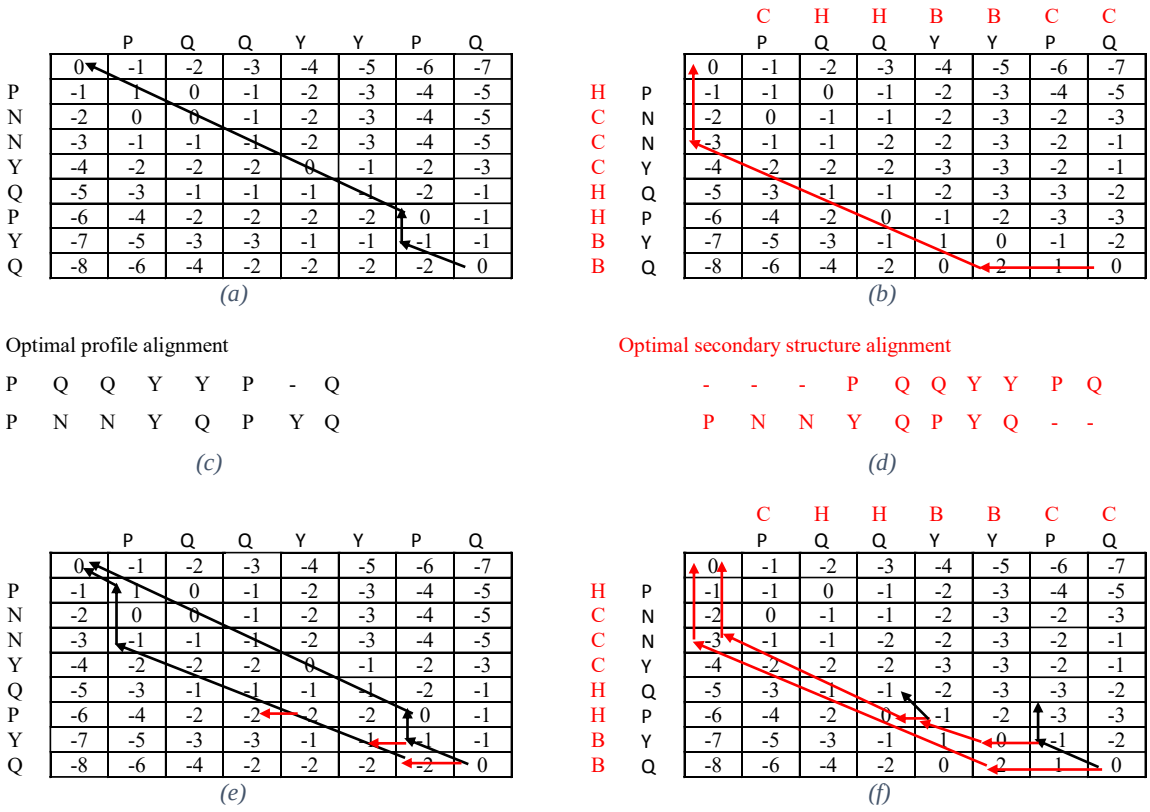


Figure 1.(a) The Needleman alignment matrix based on the profile with the maximum-match path traced to generate the optimal alignment. (b) The Needleman alignment matrix based on the secondary structure with the maximum-match path traced to generate the optimal alignment. (c), & (d) The optimal profile alignment and the optimal secondary structure alignment respectively. (e), & (f) The Needleman alignment matrix based on the profile and the Needleman alignment matrix based on the secondary structure respectively with the maximum-match path traced along with the splits due to disagreement of the other matrix, where the decisions taken based on the profile are marked on black and the ones based on the secondary structure are marked on red.

Table 1 Non-dominated alignments

Non-dominated alignment	Profile Score	Secondary Structure Score
P Q Q Y Y P - Q P N N Y Q P Y Q	0	-8
- - - P Q Q Y Y P Q P N N Y Q P Y Q - -	-6	0
- - P Q Q Y Y P Q P N N Y Q P Y - Q	-3	-3
- - - P Q Q Y Y P Q P N N Y Q P - Y - Q	-4	-2
- P - Q Q Y Y P Q P N N Y Q P Y - Q	-3	-3
- - P - Q Q Y Y P Q P N N Y Q P Y Q - -	-6	0
- - P - Q Q Y Y P Q P N N Y Q P - Y - Q	-4	-2
P - - Q Q Y Y P Q P N N Y Q P Y Q -	-1	-5
- P - - Q Q Y Y P Q P N N Y Q P Y Q - -	-6	0
- P - - Q Q Y Y P Q P N N Y Q P - Y - Q	-4	-2
P - - - Q Q Y Y P Q P N N Y Q P Y Q - -	-2	-4

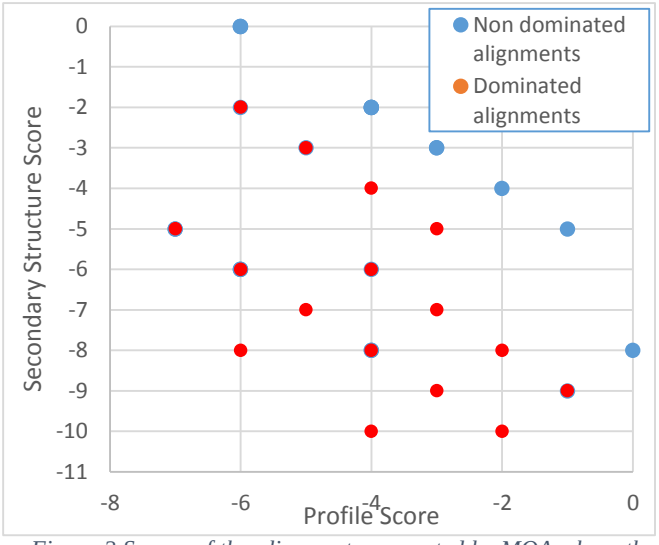


Figure 2 Scores of the alignments generated by MOA where the red ones represent the dominated alignments and the blue ones represent the non-dominated alignments

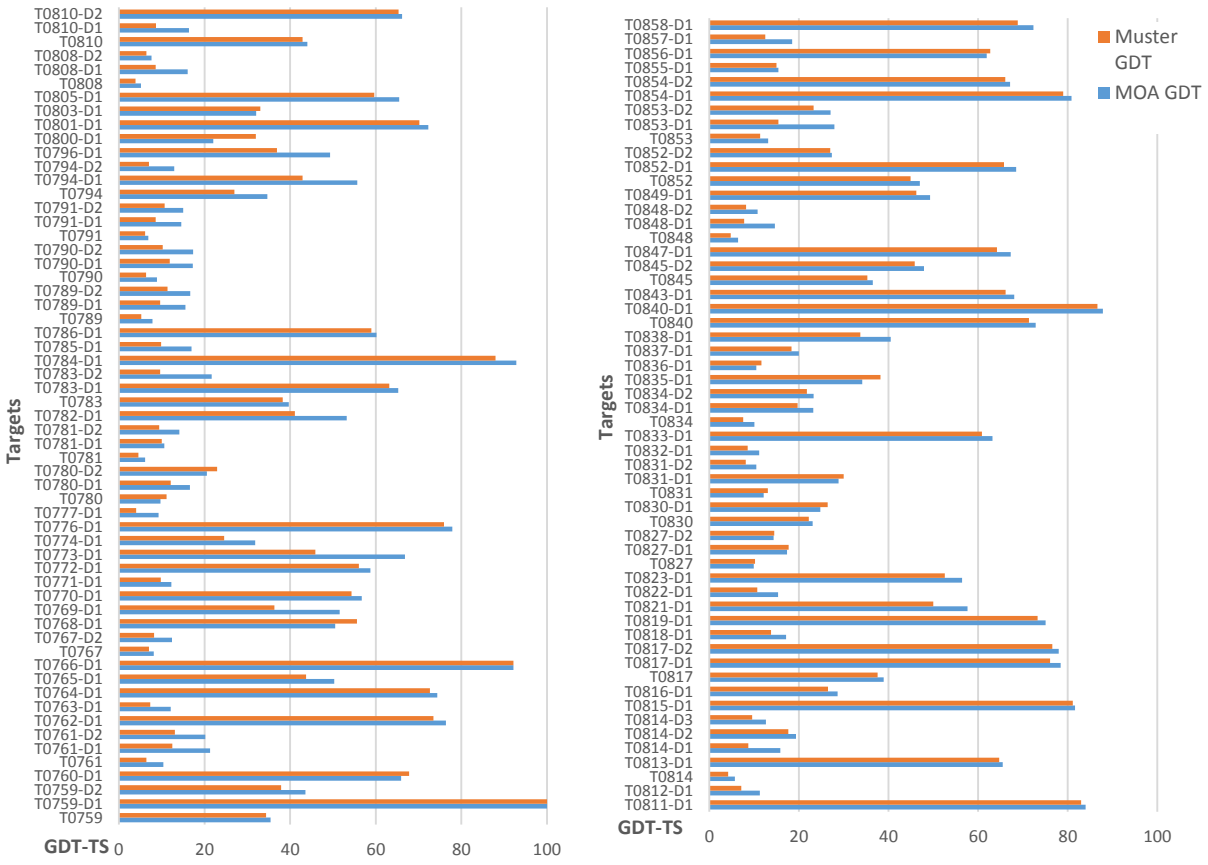


Figure 3 The GDT-TS score of Muster alignment and MOA alignment to CASP 11 targets with the top-ranked template selected by Muster. MOA achieved a higher or equal GDT-TS score for 102 targets and most of the time MOA seven of them the difference is more than 10 i.e. T0773-D1

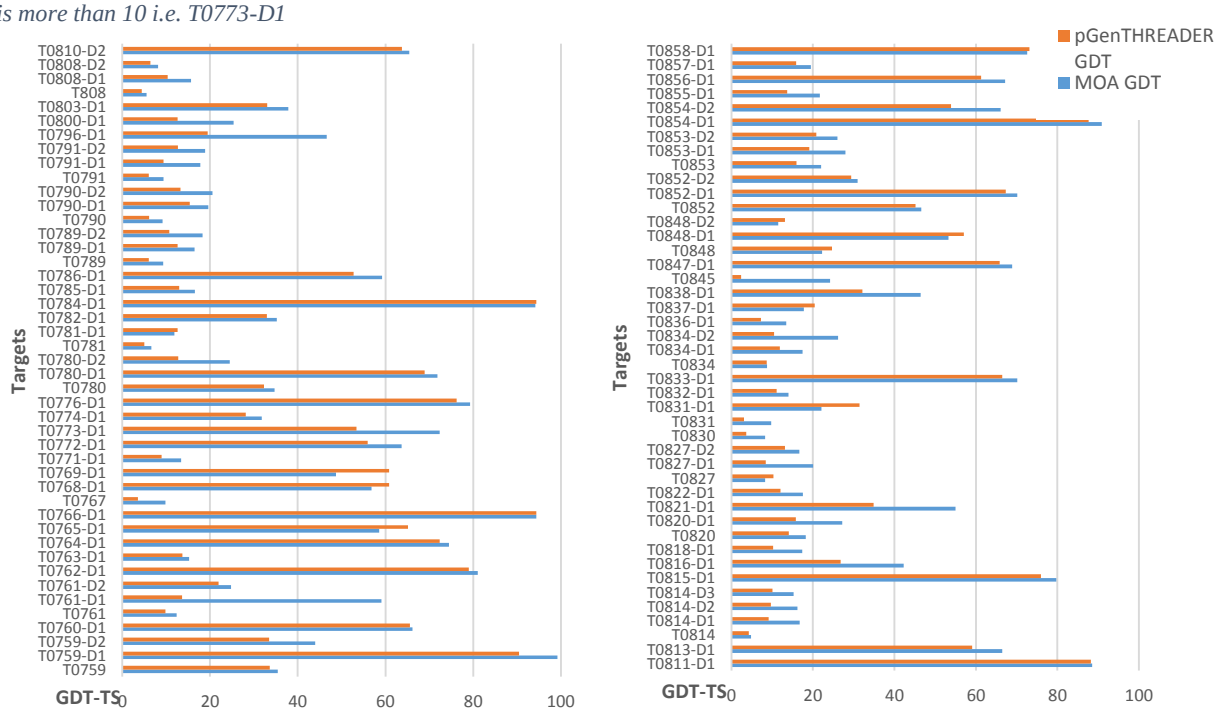


Figure 4 The GDT-TS score of pGenTHREADER alignment and MOA alignment to CASP 11 targets with the top-ranked template selected by pGenTHREADER. In 79 targets MOA GDT-TS score is higher or equal pGenTHREADER, 13 of them MOA GDT-TS score was 10 points higher than pGenTHREADER. i.e. T0840

4 Conclusions

Protein alignment is fundamental to many biological problems. Hence, we developed MOA, a multi-objective sequence alignment algorithm. There is usually a trade-off between the different objectives used in protein alignment, which makes it impossible to generate a single alignment that optimize all the objectives. Compared to finding a single alignment, MOA yields a better chance of obtaining the correct alignment and then achieving protein structure models with higher accuracy. MOA was examined on a set of CASP11 targets using the following objectives: (1) sequence profile, (2) secondary structure, and solvent accessibility objective functions. MOA has demonstrated a competitive results compared to other state of art methods.

Despite the competitive results shown from MOA, it suffers from two main deficiencies which we will work on improving in our future research. First, the MOA algorithm will generate a new alignment whenever the objectives disagree with each other, which may lead an exponential growth of the number of the traces and then end up with a large number of alignments. This is very computationally costly, particularly when aligning long protein sequences. In fact, we are only interested in the non-dominated alignments. Second, MOA generate some of the Pareto-optimal front alignments, but it doesn't guarantee the generation of the entire Pareto-optimal front.

5 Acknowledgements

This work is supported in part by the National Science Foundation under Grant No. 1066471 and the Natural Science Foundation of China under Grant No. 61728211.

6 References

[1] G. Barton, "Protein sequence alignment and database scanning," in *Protein Structure prediction - a practical approach*, Oxford, IRL Press at Oxford University Press, 1996.

[2] O. Gotoh, "Multiple sequence alignment: algorithms and applications," *Adv Biophys,* vol. 36, no. 159-206, 1999.

[3] S. B. Needleman and W. D. Christian, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of molecular biology,* vol. 48, no. 3, pp. 443-453, 1970.

[4] M. Gerstein and M. Levitt, "Using Iterative Dynamic Programming to Obtain Accurate Pairwise and Multiple Alignments of Protein Structures," in *ISMB-96*, 1996.

[5] S. F. Altschul, "Generalized Affine Gap Costs for Protein Sequence Alignment," *PROTEINS: Structure, Function, and Genetics,* vol. 32, pp. 88-96, 1998.

[6] M. Gribskov, A. D. Mclachlan and D. Eisenberg, "Profile analysis: detection of distantly related proteins," *Proc Natl Acad Sci,* vol. 84, pp. 4355-4358, 1987.

[7] S. F. Altschul, T. L. Madden, A. A. Schaffer and J. Zhang, "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs," *Nucleic Acids Research,* vol. 25, no. 17, pp. 3389-3402, 1997.

[8] J. Park, K. Karplus, C. Barrett, R. Hughey, D. Haussler, T. Hubbard and C. Chothia, "Sequence Comparisons Using Multiple Sequences Detect Three Times as Many Remote Homologues as Pairwise Methods," *J. Mol. Biol.,* vol. 284, pp. 1201-1210, 1998.

[9] M. A. Marti-Renom, M. S. Madhusudhan and A. Sali, "Alignment of protein sequences by their profiles," *Protein Science,* vol. 13, no. 4, pp. 1071-1087, 2004.

[10] R. Hughey and A. Krogh, "Hidden Markov models for sequence analysis: extension and analysis of the basic method," *Computer Appl. Biosci.,* vol. 12, no. 2, pp. 95-107, 1996.

[11] V. D. Francesco, V. Geetha, J. Garnier and P. J. Munson, "Fold recognition using predicted secondary structure sequences and hidden Markov models of protein folds," *Proteins: Struct. Funct. Genet.,* vol. 29, no. S1, pp. 123-128, 1997.

[12] K. Karplus, K. Sjolander, C. Barrett and M. Cline, "Predicting protein structure using hidden Markov models," *Proteins: Struct. Funct. Genet.,* vol. 29, no. S1, pp. 134-139, 1997.

[13] S. R. Eddy, "Profile hidden Markov models," *Bioinformatics,* vol. 14, no. 9, pp. 755-763, 1998.

[14] D. G. Higgins and P. M. Sharp, "CLUSTAL: a package for performing multiple sequence alignment on a microcomputer," *Gene,* vol. 73, pp. 237-244, 1988.

[15] J. D. Thompson, D. G. Higgins and T. J. Gibson, "CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice," *Nucleic Acids Research,* vol. 22, no. 22, pp. 4673-4680, 1994.

[16] R. C. Edgar and K. Sjolander, "SATCHMO: sequence alignment and tree construction using hidden Markov models," *Bioinformatics,* vol. 19, no. 11, pp. 1404-1411, 2003.

[17] M. S. Johnson and J. Overington, "A Structural Basis for Sequence Comparisons:: An Evaluation of Scoring Methodologies," *Molecular Biology,* vol. 233, no. 4, pp. 716-738, 1993.

[18] A. Prlic, F. Domingues and M. Sippl, "Structure-derived substitution matrices for alignment of distantly related sequences," *Protein Engineering Design and Selection,* vol. 13, no. 8, pp. 545-550, 2000.

[19] Q. Le, F. Sievers and D. G. Higgins, "Protein multiple sequence alignment benchmarking through secondary structure prediction," *Bioinformatics,* vol. 33, no. 9, pp. 1331-1337, 2017.

[20] D. Jones, W. Taylor and J. Thornton, "A new approach to protein fold recognition.," *Nature,* vol. 358, no. 6381, pp. 86-89, 1992.

[21] L. A. Kelley, R. M. MacCallum and M. J. E. Sternberg, "Enhanced Genome Annotation Using Structural Profiles in the Program 3D-PSSM," *JMB,* vol. 299, pp. 499-520, 2000.

[22] J. Shi, T. L. Blundell and K. Mizuguchi, "FUGUE: sequence-structure homology recognition using environment-specific substitution tables and structure-dependent gap penalties," *Journal of Molecular Biology,* vol. 310, no. 1, pp. 243-257, 2001.

[23] L. Rychlewski, L. Jaroszewski, W. Li and A. Godzik, "Comparison of sequence profiles. Strategies for structural predictions using sequence information," *Protein Science,* vol. 9, no. 2, pp. 232-241, 2000.

[24] J. D. Blake and F. E. Cohen, "Pairwise sequence alignment below the twilight zone," *Journal of Molecular Biology,* vol. 307, no. 2, pp. 721-735, 2001.

[25] L. Jaroszewski, L. Rychlewski and A. Godzik, "Improving the quality of twilight-zone alignments," *Protein Science,* vol. 9, no. 8, pp. 1487-1496, 2000.

[26] G. Yona and M. Levitt, "Towards a complete map of the protein space based on a unified sequence and structure analysis of all known proteins.," in *Int. Conf. Intell. Syst. Mol. Biol*, 2000.

[27] S. Wu and Y. Zhang, "LOMETS: A local meta-threading-server for protein structure prediction," *Nucleic Acids,* vol. 35, no. 10, pp. 3375-3382, 2007.

[28] S. Wu and Y. Zhang, "MUSTER: Improving protein sequence profile-profile alignments by using multiple sources of structure information," *Proteins,* p. 547–556, 2008.

[29] Y. Yang, E. Faraggi, H. Zhao and Y. Zhou, "Improving protein fold recognition and template-based modeling by employing probabilistic-based matching between predicted one-dimensional structural properties of query and corresponding native properties of templates," *Bioinformatics,* vol. 27, no. 15, pp. 2076-2082, 2011.

[30] A. Lobley, M. I. Sadowski and D. T. Jones, "pGenTHREADER and pDomTHREADER: new methods for improved protein fold recognition and superfamily discrimination," *Bioinformatics,* vol. 25, no. 14, pp. 1761-1767, 2009.

[31] Y. Li, "MOMCMC: An efficient Monte Carlo method for multi-objective," *Computers & Mathematics with Applications,* vol. 64, no. 11, pp. 3542-3556, 2012.

[32] A. Y. Y. L. W. Zhu, "DEMCMC-GPU: An Efficient Multi-Objective Optimization Method with GPU Acceleration on the Fermi Architecture," *New Generation Computing,* vol. 29, no. 2, pp. 163-184, 2011.

[33] I. R. S. C. E. J. Yaohang Li, "Improving Predicted Protein Loop Structure Ranking using a Pareto-Optimality Consensus Method," *BMC Structural Biology,* vol. 10, p. 22, 2010.

[34] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller and D. J. Lipman, "Gapped BLAST and PSI-BLAST: a new generation of protein data base search programs," *Nucleic Acids Research,* vol. 25, no. 17, pp. 3389-3402, 1997.

[35] J. Skolnick and D. Kihara, "Defrosting the frozen approximation: PROSPECTOR--a new approach to threading," *Proteins,* vol. 42, no. 3, pp. 319-331, 2001.

[36] J. Skolnick, D. Kihara and Y. Zhang, "Development and large scale benchmark testing of the PROSPECTOR_3 threading algorithm," *Proteins,* vol. 56, no. 3, pp. 502-518, 2004.

[37] H. Zhou and J. Skolnick, "Ab Initio Protein Structure Prediction Using Chunk-TASSER," *Biophysical,* vol. 93, no. 5, pp. 1510-1518, 2007.

[38] A. Yaseen and Y. Li, "Context-based features enhance protein secondary structure prediction accuracy," *Journal of chemical information and modeling,* vol. 54, no. 3, pp. 992-1002, 2014.

[39] A. Yassen and Y. Li, "CASA: A Protein Solvent Accessibility Prediction Server using Context-based Features to Enhance Prediction Accuracy," *BMC Bioinformatics,* 2014.

[40] D. T. Jones, "GenTHREADER: an efficient and reliable protein fold recog- nition method for genomic sequences," *Journal of Molecular Biology,* vol. 287, pp. 797-815, 1999.

[41] A. Š. a. T. L. Blundell., "Comparative protein modelling by satisfaction of spatial restraints.," *J. Mol. Biol.,* vol. 234, pp. 779-815, 1993.

A Two-level Scheme for Quality Score Compression

Jan Voges[+*], Ali Fotouhi[§], Jörn Ostermann[+] and M. Oğuzhan Külekci[‡*]

[+]Institut für Informationsverarbeitung, Leibniz Universität Hannover
Hannover, Germany
{voges,office}@tnt.uni-hannover.de

[§]Electronics and Communications Engineering Department, Istanbul Technical University
Istanbul, Turkey
fotouhi@itu.edu.tr

[‡]Informatics Institute, Istanbul Technical University
Istanbul, Turkey
kulekci@itu.edu.tr

Abstract

Previous studies on quality score compression can be classified into two main lines: lossy schemes and lossless schemes. Lossy schemes enable a better management of computational resources. Thus, in practice, and for preliminary analyses, bioinformaticians may prefer to work with a lossy quality score representation. However, the original quality scores might be required for a deeper analysis of the data. Hence, it might be necessary to keep them; in addition to lossy compression this requires lossless compression as well. We developed a space-efficient hierarchical representation of quality scores, QScomp, which allows the users to work with lossy quality scores in routine analysis, without sacrificing the capability of reaching the original quality scores when further investigations are required. Each quality score is represented by a tuple via a novel decomposition. The first and second dimensions of these tuples are separately compressed such that the first-level compression is a lossy scheme. The compressed information of the second dimension allows the users to extract the original quality scores. Experiments on real data reveal that the down-stream analysis with the lossy part — spending only 0.49 bits per quality score on average — shows a competitive performance, and that the total space usage with the inclusion of the compressed second dimension is comparable to the performance of competing lossless schemes.

QScomp is written in C++ and can be downloaded from https://github.com/voges/qscomp.

This work has been partially supported by The Scientific and Technological Research Council of Turkey grant number TÜBİTAK - 114E293.

* Corresponding author.

Supplementary Material can be downloaded from http://www.tnt.uni-hannover.de/~voges.

1 Introduction

Sequencing data produced by high-throughput sequencing machines are typically stored in the FASTQ format [5]. Due to the growing volumes of sequencing data, processing, transmission, and storage of the FASTQ files becomes challenging. Therefore, the compression of data stored in FASTQ files has been receiving great interest in the last years [15]. Compact representations of the data do not only help during storage and transmission by decreasing the required disk space or by enabling the possibility to better manage the available bandwidth, but also help during the analysis of the huge data volumes when the applied compression schemes support functionality such as random access over the compressed data directly. That dimension, namely compressive genomics, has been proposed and discussed in previous studies [2, 13].

FASTQ files include four lines per read. The first and the third line, beginning with the `@` and `+` symbols, respectively, indicate the read identifier and an optional description. The second line lists the read-out nucleotides. For each nucleotide in the second line a corresponding quality score (QS) Q is stored in the fourth line. The quality scores indicate the accuracy of the base-calling by $Q = -10 \cdot \log_{10} P$, where P is the error probability of the base-calling process [8].

So far, efforts in compressing raw sequencing data stored in FASTQ files have been focusing on compressing the nucleotide sequences, quality scores, and read identifiers separately. This approach yields a better performance than jointly compressing the different streams

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

since these streams have divergent statistical properties. Previous studies on quality score compression can be further separated into two categories: lossy schemes and lossless schemes. The lossy methods achieve much better compression ratios by sacrificing some information. This is done by reducing the alphabet size of the quality scores according to specific quantization methods. Although these lossy approaches help a lot in terms of storage and transmission of the data, the original values might still be required for further analyses [20].

The daily practice in sequencing data analysis starts with regular routines. In further steps of the analysis, deeper investigations are performed on the reads that are mapped to regions of interest detected by these regular routines. Quantized quality scores may work well during the initial processing unless the incorporated quantization does impact further steps significantly. Thus, when the target regions regarding the tested hypothesis become clear, necessity to access the original quality scores of the selected reads may become unavoidable during further down-stream analyses. Yet another reason to keep the original values stems from the underlying thought that the original quality scores might be required by new methods in the future. Specifically, in large population genomics projects, the owners of the data may prefer lossless compression techniques. Thus, an approach would be preferable where the users have the choice to work effectively in the first stage with quality scores represented with a lossy scheme, but at the same time have the choice to reach the original values in following analysis steps.

Motivated by this demand, we explore in this study a two-level approach for the compact representation of the quality scores. By using a novel decomposition scheme $\mathcal{D}$, we represent each quality score Q with a tuple $\mathcal{D}(Q) \rightarrow \langle q_1, q_2 \rangle$. The compression of the q_1 values constitutes the first compression level, and compressing the q_2 values creates the second level, where the q_1 values determine the context during the compression of the q_2 sequence. The first level is the lossy representation of the quality scores Q. Thus, working with this level corresponds to a lossy scheme. Given q_1 and q_2, the inverse decomposition $\mathcal{D}^{-1}$ yields the original quality scores by $Q \leftarrow \mathcal{D}^{-1}(q_1, q_2)$. This way, we preserve the capability to extract the original values. With such a two-level approach, both lossy compression and lossless compression of the quality scores can be achieved hierarchically. In the scope of this paper, we evaluate the lossy layer in terms of its effect on down-stream analyses. The space occupied by the first level and the second levels is expected to be competitive to previously proposed lossless schemes.

2 Previous Studies

In a FASTQ file the alphabet for the nucleotides (i.e., `A`, `C`, `G`, `T`, and `N`) is usually much smaller than that of the quality scores, which typically stem from an alphabet of size 40 or 41 [5]. Thus, quality scores at full resolution are in general more difficult to compress. Therefore, the overall success of compressing an input FASTQ file depends more on the representation of the quality scores than on the compression of the nucleotide sequences.

Lossless compression techniques focus on detecting regularities in quality score streams [22]. For instance, some of the quality scores are likely to be more frequent than others, or several biases may appear in some positions of the reads due to the underlying sequencing technology. Remember that a compression scheme can be viewed as a two-step process, where the first phase is to devise a context model describing the data, and the second phase is to encode the data that is represented with that model using an entropy coder. General-purpose FASTQ compressors mainly differ in their context modeling approaches. The DSRC scheme defines three models for quality score streams, and represents a given quality score sequence according to its best-fitting model [6]. SCALCE [9] and Quip [12] make use of a single standard order-3 context model, and encode every quality score according to its three immediate predecessors. Fastqz [3] applies a more complex scheme that uses relations in the near predecessors to define the context of the current quality score.

Lossy compression was considered based on the assumption that the resolution of raw quality scores is much higher than required for accuracy evaluation, and that the tools in the analysis pipelines will not be affected much from a lossy representation. It was proven that this assumption is true, and more than that, actually lossy representations improve the efficiency of down-stream analyses in many cases [23, 17]. The authors of [22] explored different binning strategies and their effects on the compression efficiency. Besides simple bucketing that uses fixed-length intervals, variable-length intervals inferred through a number of different statistical measures have also been proposed in [4]. Another statistical approach has been introduced with QualComp [16]. QualComp fits a Gaussian distribution to the quality score sequences (i.e., vectors), and provides users with the ability to define the level of acceptable distortion during encoding. According to the specified number of bits to be used per quality score, QualComp performs the optimal alteration of the quality scores such that the mean squared error is minimized according to the precomputed Gaussian model. This idea has

been further improved by the more recent QVZ and QVZ 2 compressors [14, 10]. Besides the binning and statistical inference approaches, there are other efforts which exploit the information contained in the read-out nucleotide sequences [11, 23, 21]. For example, the Quartz compressor [23] sets the quality scores of the most frequent k-mers to a predefined high value with the motivation that if a specific nucleotide sequence is observed many times, then its correctness does not need any further verification from the quality scores. Thus, the quality scores can be set to a fixed value. This way the entropy is reduced and higher compression performance is achieved.

3 Proposed Method

When an analysis pipeline automatically returns results for a set of reads (stored in a FASTQ file), the analyst usually feels the necessity to perform a verification of these results by investigating the reads together with their associated quality scores. A bioinformatician working on such reads might become suspicious when she observes low quality scores since those indicate a possible error in the base-calling process, which could have then caused problems in the automatically produced results. Similarly, when quality scores are larger than a threshold, it does not tell much to the analyst in most cases as there appears to be not much practical difference between the 99.999% accuracy with $Q = 50$ than 99.9999% with $Q = 60$. This difference becomes less and less important as long as the quality scores get higher. On the other side, due to the logarithmic nature of the quality scores, $Q = 10$ is quite different from $Q = 20$, since the first case implies 90% accuracy, while the second indicates 99% accuracy in the base-calling process.

Therefore, it seems that a simple bucketing approach with short intervals for the small quality scores and larger intervals for the higher quality scores might work well in practical analyses. Hence, we propose to decompose a quality score Q into the tuple

$$\mathcal{D}(Q) \rightarrow \langle q_1, q_2 \rangle \tag{1}$$

such that

$$q_1 = \text{round}(\sqrt{Q}), \tag{2}$$

$$q_2 = Q - \left(q_1^2 - q_1 + 1\right). \tag{3}$$

Notice that given q_1 and q_2, the inverse decomposition yields the original quality score as

$$Q = \mathcal{D}^{-1}(q_1, q_2) = q_1^2 - q_1 + 1 + q_2. \tag{4}$$

This decomposition is inpired by the representation of integers in an Elias gamma code [7] (or its generalization, the Exp-Golomb code [18]). Assume $Q = q_1^2 + c$ with $c = 1 - q_1 + q_2$. If Q is an n-bit binary number, then q_1 is an $n/2$-bit binary number and c lies in the interval $[0, 2b]$. Then q_1 can be encoded using any universal coding. Given q_1, the number of bits necessary to represent c can be determined as $\log_2(2q_1 + 1)$. However, as the scope of this work is the two-level representation of quality scores and not the exploration of sophisticated entropy coding schemes, we use the well-known general-purpose compressor bzip2 for the compression of the tuples $\mathcal{D}(Q)$.

Table 1 shows the decomposition of quality scores in the interval $[30, 43]$. The proposed decomposition creates buckets of length $(2 \cdot q_1)$, where typically $q_1 \in \{6, 7, 8, 9, 10, 11\}$ since in the FASTQ format the quality scores are between 33 and 126 (i.e., in the range of printable ASCII characters). The first $(q_1 - 1)$ of the items in a bucket are promoted to a better quality, whereas the last q_1 are faced with a penalty. Notice that the $(2 \cdot q_1)$ items long bins are relatively short for the smaller q_1 values, which fits to the motivating observation described above.

Without incorporating the q_2 values, the representation of quality scores (only by their corresponding q_1 values) creates a simple lossy scheme. In that sense, a FASTQ file in which all quality scores are changed to their q_1^2 values will exhibit a better compressibility since the alphabet for the quality scores is reduced to at most 6 symbols instead of $94 (= 126 - 33 + 1)$ possible characters. Remember that in general the observed number of symbols is around 40 as opposed to the theoretically possible 90+ symbols. Similarly, when the users would like to pertain the capability to retrieve the original scores, then they need to also keep the q_2 sequence. Instead of handling the q_2 sequence as a single stream, which would force the subsequent compressor to assume the most general alphabet for the q_2 sequence, clustering the q_2 values according to their corresponding q_1 values would improve the compression ratio (as the q_1 value in a tuple specifies the exact alphabet for the q_2 values). Thus, for each distinct q_1 value observed in the input FASTQ file, we maintain a separate sequence of q_2 values. Finally, we compress the q_1 values and the multiple q_2 sequences individually. Any general-purpose compressor can be applied. As already mentioned, we prefer to use bzip2. Surely, the users of the proposed system can proceed with different choices at this step.

4 Experimental Results

In this section we provide experimental results for the evaluation of the proposed compression scheme QScomp. We compare QScomp to three competitors, namely Crumble

Table 1: An example describing the proposed representation of quality scores.

		(q_1-1) items					q_1^2	q_1 items						
Q	30	31	32	33	34	35	**36**	37	38	39	40	41	42	43
q_1	5	6	6	6	6	6	6	6	6	6	6	6	6	7
q_2	9	0	1	2	3	4	5	6	7	8	9	10	11	0

(https://github.com/jkbonfield/crumble), Quartz [23], and QVZ 2 [10]. For the used versions and an indication of their support for lossless and lossy compression, respectively, we refer the reader to the Supplementary Material. Note that QScomp is the only tool which truly is able to operate in the lossless and in the lossy mode.

The data sets used to evaluate the performance of the selected compression tools originate from the same individual, namely NA12878. For this individual, the National Institute of Standards and Technology (NIST) released a consensus set of variants which we used for our analyses [24]. Note that similar analyses were conducted in other works [17, 1, 21]. The selected data sets are shown in Table 2. For more information on the used data sets we refer the reader to the Supplementary Material.

Table 2: Data sets selected for the evaluation.

ID	Name	Technology	Coverage
H01	ERR174324	Illumina HiSeq 2000	14×
H11	SRR1238539	Ion Torrent	10×
H12	Garvan replicate	Illumina HiSeq X	49×

Moreover, for the evaluation of the proposed compression scheme QScomp, we selected three different variant calling pipelines. The first pipeline is composed of GATK [20] variant calling (using the HaplotypeCaller tool) and SNP extraction with subsequent filtering of variants using GATK Vector Quality Score Recalibration (VQSR) with four different filter values. The second pipeline is also composed of GATK variant calling using the HaplotypeCaller tool and SNP extraction but followed by the more traditional hard filtration of variants instead of VQSR. The third pipeline uses Platypus [19] for variant calling. For the individual commands and tools and auxiliary files used, we refer the reader to the Supplementary Material.

Each of the mentioned pipelines outputs a set of variants in the VCF file format. Subsequently, each set of variants is compared to the consensus set of variants. We perform this comparison using the tool hap.py (https://github.com/Illumina/hap.py) released by Illumina and adopted by the Global Alliance for Genomics and Health (GA4GH). This benchmarking tool outputs the following values for each comparison:

- True Positives (T.P.): All those variants that are both in the consensus set and in the set of called variants.
- False Positives (F.P.): All those variants that are in the called set of variants but not in the consensus set.
- False Negatives (F.N.): All those variants that are in the consensus set but are not in the set of called variants.
- Non-Assessed Calls: All those variants that fall outside of the consensus regions defined by a BED file.

These values are used to compute the following two metrics:

- Recall/Sensitivity: This is the proportion of called variants that are included in the consensus set; that is, $R = {}^{\text{T.P.}}/_{(\text{T.P.} + \text{F.N.})}$,
- Precision: This is the proportion of consensus variants that are called by the variant calling pipeline; that is, $P = {}^{\text{T.P.}}/_{(\text{T.P.} + \text{F.P.})}$.

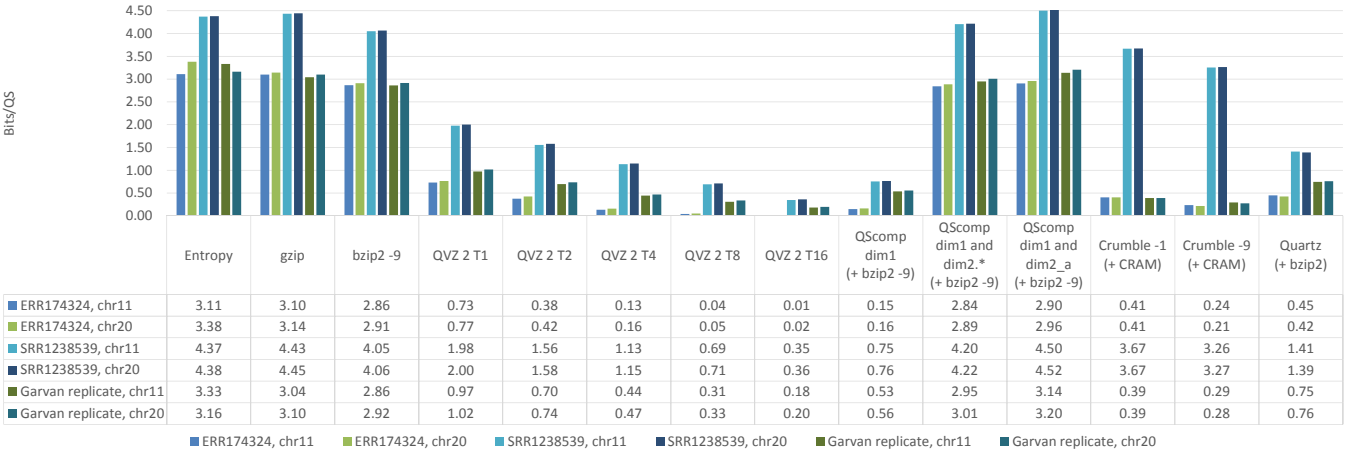
Finally, we measured the maximum memory usage and the execution time of each tool on each dataset with GNU time.

4.1 Performance Analysis of the Proposed Scheme

In this section we first show the compression ratios of all tools and for all datasets from Table 2.

Figure 1 shows the compression results for all tools in bits per quality score. In addition to the compression results for the mentioned tools, we also show the memoryless entropy per original quality score, which is 3.62 bits per quality score, averaged over all data sets. Furthermore, we show the gzip and bzip2 compression results for the raw quality scores, which are 3.54 bits per quality score and 3.27 bits per quality score, also averaged over all data sets.

As shown in Figure 1, the lossy quality score representation obtained using QScomp with subsequent bzip2 compression (i.e., "QScomp dim1 (+ bzip2 -9)") yields 0.49 bits per quality score on average. This



	Entropy	gzip	bzip2 -9	QVZ 2 T1	QVZ 2 T2	QVZ 2 T4	QVZ 2 T8	QVZ 2 T16	QScomp dim1 (+ bzip2 -9)	QScomp dim1 and dim2.* (+ bzip2 -9)	QScomp dim1 and dim2_a (+ bzip2 -9)	Crumble -1 (+ CRAM)	Crumble -9 (+ CRAM)	Quartz (+ bzip2)
ERR174324, chr11	3.11	3.10	2.86	0.73	0.38	0.13	0.04	0.01	0.15	2.84	2.90	0.41	0.24	0.45
ERR174324, chr20	3.38	3.14	2.91	0.77	0.42	0.16	0.05	0.02	0.16	2.89	2.96	0.41	0.21	0.42
SRR1238539, chr11	4.37	4.43	4.05	1.98	1.56	1.13	0.69	0.35	0.75	4.20	4.50	3.67	3.26	1.41
SRR1238539, chr20	4.38	4.45	4.06	2.00	1.58	1.15	0.71	0.36	0.76	4.22	4.52	3.67	3.27	1.39
Garvan replicate, chr11	3.33	3.04	2.86	0.97	0.70	0.44	0.31	0.18	0.53	2.95	3.14	0.39	0.29	0.75
Garvan replicate, chr20	3.16	3.10	2.92	1.02	0.74	0.47	0.33	0.20	0.56	3.01	3.20	0.39	0.28	0.76

Figure 1: Compression ratios results.

result is comparable to the results obtained with QVZ 2 when a target mean squared error (MSE) of 8 (i.e., "QVZ 2 T8") is specified, which yields 0.35 bits per quality score on average.

We can observe from the figure that the lossless quality score representation of QScomp with subsequent bzip2 compression (i.e., "QScomp dim1 and dim2.* (+ bzip2 -9)") is capable of delivering 3.35 bits per quality score, which is slightly below the entropy, as expected. The two-level scheme of QScomp with conditional compression of the second level with respect to first level is slightly superior to just compressing the quality scores with gzip, and comparable to compressing the quality scores with bzip2. Thus, QScomp does not sacrifice the lossless compression performance, while combining the lossless and lossy compression via its unique two-level scheme. We finally show in Figure 1 the results of compressing the joint single sequence of q_2 values (i.e., "QScomp dim1 and dim2_a (+ bzip -9)"). This experiment yields 3.53 bits per quality score. This results suggests that the proposed separate compression of multiple q_2 sequences is superior to just compressing the q_2 residues as a single stream.

Furthermore, in the Supplementary Material we show the maximum RAM usage and the running times, respectively, of all tools. QScomp exhibits the least RAM usage of all tools, with 3.4 MB on average, due to its low algorithmic complexity. The running times of QScomp are comparable to that of the different runs of QVZ 2 and even two orders of magnitude lower than that of Quartz.

4.2 Variant Calling Results

In this section, we show the results of variant calling with the GATK + VQSR pipeline. For further results obtained from running the other two pipelines, we refer the reader to the Supplementary Material. For the first set of simulations we used the paired-end run ERR174324 of the NA12878 individual. This run was sequenced by Illumina on an Illumina HiSeq 2000 system as part of their Platinum Genomes project. The coverage of this data set is 14×. Due to the size of the data and the following the approach of [17] we consider chromosomes 11 and 20. Furthermore, we averaged the Recall and Precision metrics over the two chromosomes (11 and 20) and the four VQSR filter values ($\theta \in \{90, 99, 99.9, 100\}$) which yields 2 plots. In what follows, we did the same for the other data sets. Thus, we present in total 6 plots (i.e., 3 data sets × 2 metrics) in this section.

We can observe from Figure 2 that QScomp compresses the quality scores down to 0.16 bits per quality score while the Precision is retained. However, we also observe a slight drop in Recall, compared to the results for the uncompressed data.

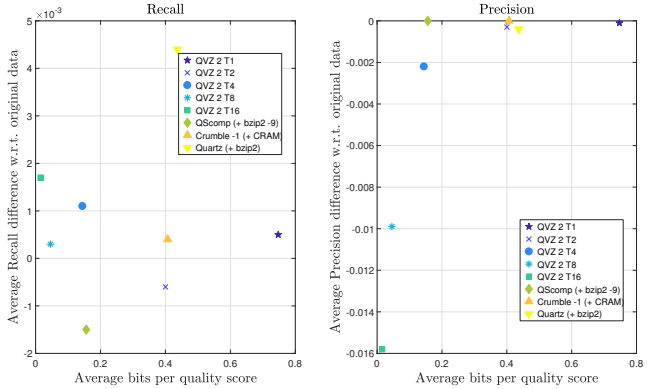


Figure 2: Recall and Precision results averaged over both chromosomes (11 and 20) and all four VQSR filter values for the Illumina HiSeq 2000 data set (ERR174324) with a coverage of 14×.

Next, we show the results for the SRR1238539 run on the NA12878 individual for which an Ion Torrent sequencing machine was used. The coverage of this data set is 10×. Again, chromosomes 11 and 20 were considered due to the size of the data. Moreover, the results shown are also the results of averaging over the same four filter values and both chromosomes. Figure 3 shows that QScomp is the worst performer in terms of both Recall and Precision. Since all other tools exhibit a similar performance, we must conclude that the assumptions used for the construction of the binning scheme implemented in QScomp do not seem to hold for the quality score statistics produced by Ion Torrent sequencing machines.

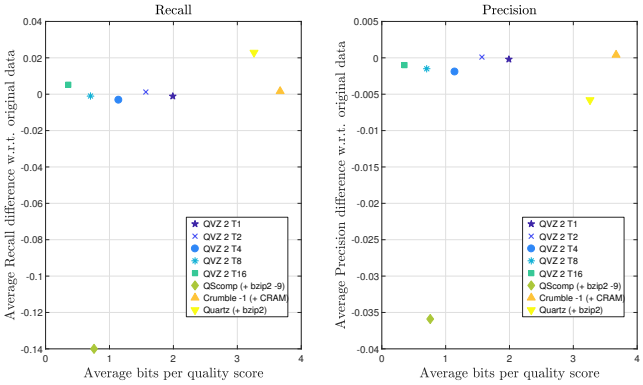


Figure 3: Recall and Precision results averaged over both chromosomes (11 and 20) and all four VQSR filter values for the Ion Torrent data set (SRR1238539) with a coverage of 10×.

Finally, we used the first replicate of the sample data set generated by the Garvan Institute from the Coriell Cell Repository NA12878 reference cell line. These data were sequenced on a single lane of an Illumina HiSeq X machine. The coverage of this data set is 49×. These results are shown in Figure 4. In terms of Recall and Precision, QScomp exhibits a similar performance as for the data set ERR174323, which is shown in Figure 2. Again, the Precision is retained. However, for this data set, a better Recall can be observed for all tools, including QScomp. Due to the high coverage of this data set, the competing tools are able to spend less bits per quality score than QScomp. Nevertheless, QScomp compresses the quality scores down to 0.55 bits per quality score, yielding a compression factor of 5.9 with respect to the entropy of the uncompressed data.

5 Conclusions

We presented a hierarchical quality score compression scheme, which represents the quality scores in two levels. The first level maps each quality score to its

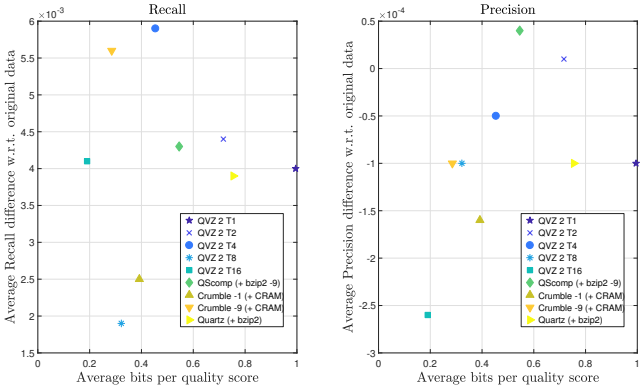


Figure 4: Recall and Precision results averaged over both chromosomes (11 and 20) and all four VQSR filter values for the Illumina HiSeq X data set (Garvan replicate) with a coverage of 49×.

nearest square integer, and the second level encodes the distance of the original quality score to its mapped value. The impact of the lossy representation of quality scores on down-stream analyses was investigated using three different variant calling pipelines. For data produced by Illumina sequencing machines, the down-stream analysis results are competitive to the results obtained with competing lossy quality score compressors. Here, the Precision is retained, while a slight drop in Recall was observed. When this lossy level is accompanied by the second level, we observe that the compression ratio is around the entropy of the original quality scores. This shows that the suggested method to represent each quality score by a tuple does not have a negative effect on the lossless compression ratio performance. What is more, we showed that the proposed separate compression of multiple second level streams is superior to the compression of the second level as a single stream. Hence, the incorporation of other quantization strategies from previous works into the proposed two-level scheme might be a reasonable future research avenue. Besides the compression ratios, the memory consumption and the running times are also important parameters. Here, with an average of only approximately 3.4 MB, QScomp shows a significant reduction in peak memory usage, and achieved the highest speed in the benchmark.

Previous studies on quality score compression proposed solutions that are either lossless or lossy. Thus, if a user prefers lossy compression, the possibility to extract the original quality scores disappears, and in the reverse case, the user looses the capability to work with lossy quality scores to reduce the necessary computing resources. The QScomp scheme introduced in this study is unique in terms of providing lossless and lossy compression in a single framework by utilizing a

hierarchical two-level representation.

In daily practice, we suggest to replace the quality scores in FASTQ files with the proposed first-level values, and to perform initial explorations with this lightweight presentation. The second-level values could for example be stored in an archive, and when deeper investigations are required the original quality scores could be retrieved.

References

[1] Claudio Alberti et al. An Evaluation Framework for Lossy Compression of Genome Sequencing Quality Values. In *2016 Data Compression Conference (DCC)*, pages 221–230, 2016.

[2] Bonnie Berger, Noah M Daniels, and Y William Yu. Computational Biology in the 21st Century: Scaling with Compressive Algorithms. *Communications of the ACM*, 59(8):72–80, 2016.

[3] James K Bonfield and Matthew V Mahoney. Compression of FASTQ and SAM Format Sequencing Data. *PloS ONE*, 8(3):e59190, 2013.

[4] Rodrigo Cánovas, Alistair Moffat, and Andrew Turpin. Lossy compression of quality scores in genomic data. *Bioinformatics*, 30(15):2130–2136, 2014.

[5] Peter J A Cock et al. The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. *Nucleic Acids Research*, 38(6):1767–1771, 2010.

[6] Sebastian Deorowicz and Szymon Grabowski. Compression of DNA sequence reads in FASTQ format. *Bioinformatics*, 27(6):860–862, 2011.

[7] Peter Elias. Universal Codeword Sets and Representations of the Integers. *IEEE Transactions on Information Theory*, 21(2):194–203, 1975.

[8] B Ewing and P Green. Base-calling of automated sequencer traces using phred. II. Error probabilities. *Genome Research*, 8(3):186–194, 1998.

[9] Faraz Hach, Ibrahim Numanagić, Can Alkan, and S Cenk Sahinalp. SCALCE: boosting sequence compression algorithms using locally consistent encoding. *Bioinformatics*, 28(23):3051–3057, 2012.

[10] Mikel Hernaez, Idoia Ochoa, and Tsachy Weissman. A Cluster-Based Approach to Compression of Quality Scores. In *2016 Data Compression Conference (DCC)*, pages 261–270, 2016.

[11] Lilian Janin, Giovanna Rosone, and Anthony J Cox. Adaptive reference-free compression of sequence quality scores. *Bioinformatics*, 30(1):24–30, 2014.

[12] Daniel C Jones, Walter L Ruzzo, Xinxia Peng, and Michael G Katze. Compression of next-generation sequencing reads aided by highly efficient de novo assembly. *Nucleic Acids Research*, 40(22):e171, 2012.

[13] Po-Ru Loh, Michael Baym, and Bonnie Berger. Compressive genomics. *Nature Biotechnology*, 30(7):627–630, 2012.

[14] Greg Malysa et al. QVZ: lossy compression of quality values. *Bioinformatics*, 31(19):3122–3129, 2015.

[15] Ibrahim Numanagić et al. Comparison of high-throughput sequencing data compression tools. *Nature Methods*, 13(12):1005–1008, 2016.

[16] Idoia Ochoa et al. QualComp: a new lossy compressor for quality scores based on rate distortion theory. *BMC Bioinformatics*, 14:187, 2013.

[17] Idoia Ochoa et al. Effect of lossy compression of quality scores on variant calling. *Briefings in Bioinformatics*, 18(2):183–194, 2016.

[18] Jörn Ostermann et al. Video coding with H.264/AVC: tools, performance, and complexity. *IEEE Circuits and Systems Magazine*, 4(1):7–28, 2004.

[19] Andy Rimmer et al. Integrating mapping-, assembly- and haplotype-based approaches for calling variants in clinical sequencing applications. *Nature Genetics*, 46(8):912–918, 2014.

[20] Geraldine A Van der Auwera et al. From FastQ data to high-confidence variant calls: the Genome Analysis Toolkit best practices pipeline. *Current Protocols in Bioinformatics*, 43(11):11.10.1–11.10.33, 2013.

[21] Jan Voges, Jörn Ostermann, and Mikel Hernaez. CALQ: compression of quality values of aligned sequencing data. *Bioinformatics*, btx737, 2017.

[22] Raymond Wan, Vo Ngoc Anh, and Kiyoshi Asai. Transformations for the compression of FASTQ quality scores of next-generation sequencing data. *Bioinformatics*, 28(5):628–635, 2012.

[23] Y William Yu, Deniz Yorukoglu, Jian Peng, and Bonnie Berger. Quality score compression improves genotyping accuracy. *Nature Biotechnology*, 33(3):240–243, 2015.

[24] Justin M Zook et al. Extensive sequencing of seven human genomes to characterize benchmark reference materials. *Scientific Data*, 3(160025), 2016.

Minimising the Deep Coalescence

Dawid Dąbkowski and Paweł Górecki

Faculty of Mathematics, Informatics and Mechanics, University of Warsaw, Poland
Warsaw, mazowieckie, 00-927, Poland
dd345130@students.mimuw.edu.pl, gorecki@mimuw.edu.pl

Abstract

Metagenomic studies identify the species present in an environmental sample usually by using procedures that match molecular sequences, e.g., genes, with the species taxonomy. Here, we formulate the problem of gene-species matching in the parsimony framework using phylogenetic gene and species trees under the deep coalescence cost and the assumption that each gene is paired uniquely with one species. In particular, we solve the problem in the cases when one of the trees is *caterpillar*. Next, we generalize the solution and propose several heuristic algorithms. Finally, we present the results of computational experiments on simulated and empirical datasets.

keywords: deep coalescence, metagenomics, species taxonomy, gene tree

1 Introduction

One of the primary goals of metagenomic studies is to identify the species present in an environmental sample. Such identification from metagenomics data is usually computationally demanding and requires complex workflows in which molecular sequences identified in the sample, i.e., reads, genes or contigs, can be matched with the species taxonomy. The gene-species matching can be expressed by using a partially labeled phylogenetic tree, where a gene tree inferred from a set of homologous sequences extracted from the sample is matched with the known species taxonomy. Here, we formulate the problem in the parsimony framework using gene and species trees under the deep coalescence model and the assumption that each gene is paired uniquely with one species.

Deep coalescence is a major process that can lead to a discordance between a gene tree and its species tree. It occurs when the time at which lineages of alleles coalesce, predates speciation events of the alleles' species [1, 2]. The discordance caused by deep coalescence can be measured by using *the deep coalescent cost* which, given two labeled trees, is efficiently computable in linear time [3]. Consequently, the cost has been studied in the context of classical problems in computational biology, e.g., gene tree parsimony [4–7], tree reconciliation [8], error correction [9] or tree rooting [10, 11].

In this article, we analyze the deep coalescence cost for a pair of bijectively labeled gene and species trees. We investigate into gene-species matching problem expressed as the *minimisation problem*, that is, *given two unlabeled trees find bijective leaf-labelings for these trees that minimise the deep coalescence cost.* While several variants of the dual maximising problem can be solved in polynomial time [12–14], little is known about the minimising problem. The closest is the minimisation problem for the general leaf-labelings, i.e., without the requirement of bijectivity. Usually, such a problem can be solved by a dynamic programming in polynomial time [15–17].

In this article, we solve the gene-species problem for bijectively labeled leaves in the cases when one of the trees is *caterpillar* by using two tree ordering operations. Next, we generalize the solution and propose several heuristic algorithms the problem for any gene and species tree topology. Finally, we present the results of computational experiments on simulated and empirical datasets.

2 Basic definitions

A *tree* in this article is a rooted binary tree $T = \langle V(T), E(T)\rangle$ such that the edges of T are directed towards leaves, i.e., if $\langle v, w\rangle \in E(T)$ then v is the parent of w. The edges incident to the *root* are called *top*. By $T(v)$ we denote a subtree of T rooted at a node v. A cluster of v, denoted $C_T(v)$, is the set of all leaves of $T(v)$. By $|T|$ we denote the *size* of T, that is, the number of its leaves. By $h(T)$ we denote the *height* of T, i.e., the maximal number of edges on the path from the root to a leaf of T. If v is a non-root node, then v_P is the parent of v and v_S is the sibling of v.

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a fixed set of $n > 1$ taxa.

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

A *labeled tree* over X is a tree having exactly n leaves bijectively labeled by the elements from X. A labeled tree is often *ordered*. In such a case, each internal node v has the left and the right child, denoted v_L and v_R, respectively. An ordered tree T induces a *labeling* $\Lambda_T\colon [n] \to X$,[1] such that $\Lambda_T(1), \Lambda_T(2), \ldots, \Lambda_T(n)$ are the taxons obtained from the leaves by the left to right traversal of T. A labeling that satisfies $\Lambda_T(i) = x_i$, for $i \in [n]$ will be called *simple*. For a node v, the subtree $T(v)$ has the labeling inherited from T.

Each edge $e \in E(T)$ can be uniquely identified by the child, therefore, in notation, we often use an edge and its terminating node (the child) alternatively. For instance, the subtree $S(v)$ can be denoted as $S(e)$ if $e = \langle v_P, v\rangle$.

In computational biology, we recognize two types of trees: a *gene tree* and a *species tree*. In this article, they are both labeled trees over the same set of taxa. Now we introduce the least common ancestor mapping, in short, LCA-mapping, from a gene tree G to a species tree S. An example is depicted in Fig. 1.

Definition 1. LCA-*mapping from a gene tree G to a species tree S is a function* $\mathsf{LCA}_S\colon G \to S$ *such that for a node v of G,* $\mathsf{LCA}_S(v)$ *is the lowest node s of S, such that each taxon from $G(v)$ is present in $S(s)$.*

Based on the LCA-mapping we can embed a gene tree G into S by mapping every edge $\langle v, w\rangle$ of G to the path in S whose endpoints are $\mathsf{LCA}_S(v)$ and $\mathsf{LCA}_S(w)$. The edges of such paths are called *lineages*. Embeddings can be visualized in the form depicted in Fig. 1. If both trees are equal, then the LCA-mapping is a bijection, and an edge is bijectively mapped to an edge. Otherwise, the embedding has some number of *extra lineages* present on edges of the species tree. For an edge $e \in E(S)$, the number, denoted $\mathsf{xl}(G, e)$, can be defined formally as [13]

$$\mathsf{xl}(G, e) = |C_S(e)| - c_e - 1,$$

where c_e is the number of internal nodes of G that are mapped to nodes of $S(e)$.

Having this, we define the deep coalescence cost.

Definition 2. *For a gene tree G and a species tree S the deep coalescence cost is defined as* $\mathsf{dc}(G, S) = \sum_{e\in E(S)} \mathsf{xl}(G, e)$.

Equivalently, it can be shown that $\mathsf{dc}(G, S) = \sum_{\langle v,w\rangle\in E(S)} \|\mathsf{LCA}_S(v), \mathsf{LCA}_S(w)\| - 1$, where $\|s, s'\|$ denote the number of edges on the shortest path connecting s and s'.

Now, we will investigate into the minimal deep coalescence cost for fixed tree topologies. For a given tree T (labeled or not) by $\mathbb{S}(T)$ we denote the set of all labeled trees T' over X such that $V(T) = V(T')$ and $E(T) = E(T')$. In other words, the elements of $\mathbb{S}(T)$ share the tree topology.

[1] $[n] = \{1, 2, \ldots, n\}$

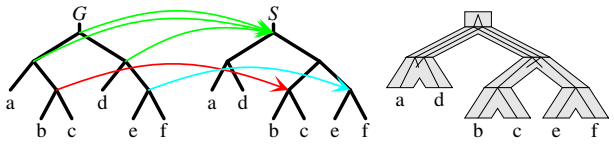


Figure 1: *Left:* An example of a gene tree G and a species tree S over $X = \{a, b, c, d, e, f\}$ with the LCA-mappings of internal nodes of G. *Right:* The embedding, or evolutionary scenario [18], explains the differences between G and S by drawing G inside S. Here, $\mathsf{dc}(G, S) = 2$ as each top edge of S has one extra lineage.

Problem 1 (MinDC)**.** *Given trees G and S. Find the minimal* $\mathsf{dc}(G', S')$*, denoted* $\widetilde{\mathsf{dc}}(G, S)$*, in the set of all pairs $\langle G', S'\rangle$ from $\mathbb{S}(G) \times \mathbb{S}(S)$.*

From the practical point of view, the most critical problem is to infer the minimal labelings, which encode the gene-species mappings. This can be expressed by seeking for the optimal gene tree as follows.

Problem 2 (Gene-Species Matching)**.** *Given a tree G and a species tree S. Find $G^* \in \mathbb{S}(G)$ such that* $\mathsf{dc}(G^*, S) = \widetilde{\mathsf{dc}}(G, S)$.

3 Results

In this section, we show how to solve our problems when one of the trees is a caterpillar, i.e., the maximum-height tree $\mathcal{C}_n$.

3.1 Caterpillar species tree

We say that an ordered tree T is size-*ordered* if for each internal node v we have $|T(v_L)| \leq |T(v_R)|$.

Theorem 1. *Given a* size-*ordered gene tree G and a* size-*ordered species tree $\mathcal{C}_n$. If both are simply labeled then* $\widetilde{\mathsf{dc}}(G, \mathcal{C}_n) = \mathsf{dc}(G, \mathcal{C}_n)$*. Furthermore,*

$$\widetilde{\mathsf{dc}}(G, \mathcal{C}_n) = \widetilde{\mathsf{dc}}(G_L, \mathcal{C}_{|G_L|}) + \widetilde{\mathsf{dc}}(G_R, \mathcal{C}_{|G_R|}) + |G_L| - 1, \quad (1)$$

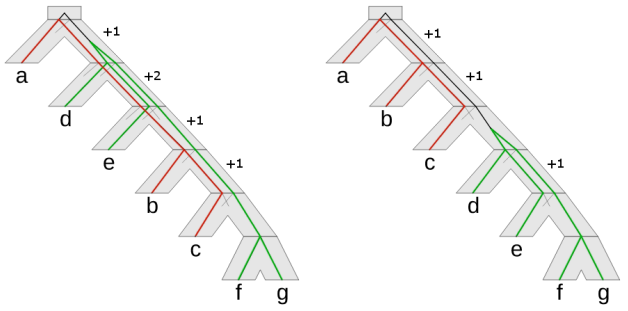
where G_L and G_R are the left and the right subtree of G, respectively.

Proof. Without loss of generality, we assume that the labeling of G is simple. Let us consider any labeling of $\mathcal{C}_n$. First, we embed G_L and G_R separately, and then,

we join them by embedding the top edges. We can write that,

$$\mathsf{dc}(G, \mathcal{C}_n) = \mathsf{dc}(G_L, \mathcal{C}_{|G_L|}) + \mathsf{dc}(G_R, \mathcal{C}_{|G_R|}) + K, \quad (2)$$

where $K > 0$ is the number of additional extra lineages from the top edges and the intersection of subtrees in the embedding. Here, the labelings of $\mathcal{C}_{|G_L|}$ and $\mathcal{C}_{|G_R|}$ are inherited from $\mathcal{C}_n$. An example is depicted in Fig. 2.



$K = 4$, $\mathsf{dc}(G, S_1) = 5$ $\quad$ $K = 2$, $\mathsf{dc}(G, S_2) = 3$

Figure 2: Embeddings of a simple labeled gene tree $G = ((a,(b,c)),((d,e),(f,g)))$ into $\mathcal{C}_n$ with two labelings. The lineages of $G_L = (a,(b,c))$ are red, $G_R = ((d,e),(f,g))$ are green, while the lineages of top edges of G are black. The contribution to dc is marked next to edges. The right embedding is optimal, as the tree is simple labeled and the lineages of G_L and G_R are disjoint.

We show that the best labeling of $\mathcal{C}_n$ is simple. The proof is by induction on the size of T. For $n = 1$ it is trivial. Assume that the statement holds true for trees of the size smaller than n. We want to minimize the value of $\mathsf{dc}(G, \mathcal{C}_n)$.

Let Λ be the labeling of the species tree $\mathcal{C}_n$ and let $m = |G_L| \leq |G_R|$. Let A and the set of indices of taxons mapped to $\mathcal{C}_n$ from G_L [2]. If Λ is simple, then $A = [m]$ and $K = m-1$ as the lineages of G_L and G_R are disjoint and there are $m-1$ lineages of the top edges (see Fig. 2). In general, for any A, let $0 = l_0 < l_1 < l_2 < \ldots < l_k = n$ be the maximal sequence such that for $1 \leq j < k$ either $l_j \in A \wedge l_j + 1 \notin A$ or $l_j \notin A \wedge l_j + 1 \in A$. E.g. for the left tree from Fig. 2, $k = 4$, $n = 7$, $l_1 = 1$, $l_2 = 3$, $l_3 = 5$ and $l_4 = 7$. Now we have $[n]$ split into k parts $P_j := \{l_{j-1} + 1, \ldots, l_j\}$ for $j \in [k]$. We can imagine a tree G_j as a tree contracted to the set of taxons from $\Lambda[P_j]$. When embedding G into $\mathcal{C}_n$, we can inductively embed G_L (and similarly G_R) as proposed in formula (2). This can be done by using every second G_j's tree and calculating only additional extra lineages in embeddings of G_j and G_{j+2} that are separated by the embedding of G_{j+1}. Including the lineages of top edges, to calculate K we have the following observations. Let s_j be the LCA-mapping of the root of G_j. For every $3 \leq j \leq k$, G_j has to be connected with G_{j-2}, which requires $|G_{j-1}| - 1$ lineages, located on the path connecting s_{j-1} with the parent of s_j, shared with the lineages of G_{j-1}. Similarly, G_2 needs $|G_1| - 1$ lineages between the parent of s_2 and the root of S. Next, if $2 \leq j < k$, there is one more lineage, i.e., the edge whose terminating node is s_j, shared with the lineages connecting G_{j-1} and G_{j+1}. We have that $|G_j| = l_j - l_{j-1}$, hence $K = k - 2 + \sum_{j=2}^{k}(|G_{j-1}| - 1) = l_{k-1} - 1 \geq \min(m, n-m) - 1 = m - 1$. So, for every labeling of $\mathcal{C}_n$, K is bounded, and this boundary is achieved only when $k = 2$. By the inductive assumption, this statement joined with the previous observations, completes the proof. □

The next theorem shows that to compute the minimal dc for the caterpillar species tree, it is sufficient to order the gene tree by size.

Theorem 2. *If G is a* size*-ordered gene tree then*

$$\widetilde{\mathsf{dc}}(G, \mathcal{C}_n) = \sum_{e \in \mathsf{Lft}(G)} |G(e)| - 1, \quad (3)$$

where $\mathsf{Lft}(G)$ is the set of all edges in G that connect a node with its left child.

Proof. It follows immediately from Thm. 1. □

Note that we cannot fully classify the minimal cost trees by writing that for a gene tree G, $\widetilde{\mathsf{dc}}(G, \mathcal{C}_n) = \mathsf{dc}(G, \mathcal{C}_n)$ if and only if G is a size-ordered tree and $\Lambda_G = \Lambda_{\mathcal{C}_n}$. This statement does not hold in general, e.g., we can swap leaves of f and g in the left species tree from Fig. 2 and the minimal cost will be preserved.

The compact formula (3), or the recursive formula (1), allows us to compute $\widetilde{\mathsf{dc}}$ in linear time. Now let us also fix the topology of G to be a *complete balanced* tree of height h, i.e., a tree of the size 2^h with all 2^h leaves on the same depth, and calculate deep coalescence cost. We have: $\widetilde{\mathsf{dc}}(\mathcal{B}_n, \mathcal{C}_n) = \sum_{i=1}^{h} 2^{i-1} \cdot (2^{h-i} - 1) = \frac{1}{2}(\sum_{i=1}^{h} 2^h - 2^i) = \frac{1}{2}(h \cdot 2^h - 2 \cdot \frac{1-2^h}{1-2}) = \frac{1}{2}(n \log_2 n + 2(1-n)) = \frac{n}{2}(\log_2 n - 2) + 1$

It shows that $\widetilde{\mathsf{dc}}$ between a complete binary and a caterpillar tree is $\sim \frac{n}{2} \log_2 n$. Having this, one may conjecture, that the maximal $\widetilde{\mathsf{dc}}$ for any gene tree versus a caterpillar species tree, both of the size n, is $\sim \frac{n}{2} \log_2 n$.

3.2 Caterpillar gene tree

In this Section, we show how to solve our Problems in the case when a gene tree is a caterpillar. The solution is

[2] Formally, $A = \Lambda^{-1}(\Lambda_G[\{1, 2, \ldots, m\}])$.

similar to the previous case, with the difference that we need a new type of order. For a node $v \in V(T)$ a *saving* of v, denoted $\mathsf{sav}(v)$, is defined recursively: $\mathsf{sav}(v) = \max(\mathsf{sav}(v_L), \mathsf{sav}(v_R)) + |T(v)| - 1$, where $\mathsf{sav}(v) = 0$ if v is a leaf. We say that a tree is sav-ordered, if, for every internal node v we have $\mathsf{sav}(v_L) \leq \mathsf{sav}(v_R)$. An example of a sav-ordered tree is depicted in Fig. 3.

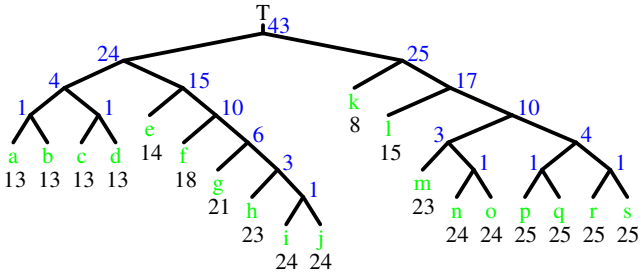


Figure 3: An example of a sav-ordered tree T with the decoration showing the values of sav for the internal nodes, and the weights $\omega_i(T)$ each leaf. This is one of the two smallest trees (to obtain the second one replace $((a,b),(c,d))$ by $(a,(b,(c,d)))$) showing that sav-ordeding is significantly different than size-ordering. Furthermore, this example shows that sav-ordering cannot be replaced by a potentially simpler ordering based on the height of subtrees.

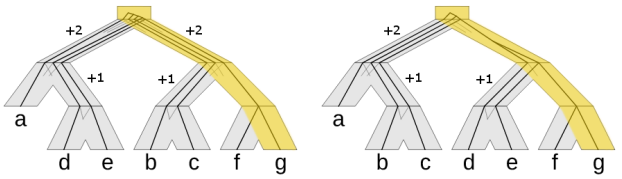
Let $E_i(T)$ denote the set of edges on the path connecting the root of T with the i-th leaf. Let $\omega_T(i) = \sum_{e \in E_i(T)} |T(e)| - 1$ be the *weight* of the i-th path. First, we show that, for a sav-ordered tree T, $\omega_i(T)$ is maximized by the rightmost path.

Lemma 1. *For any* sav*-ordered tree* T, $\max_i \omega_T(i) = \mathsf{sav}(\mathsf{root}(T)) - |T| + 1$. *Moreover, the maximum is reached by the rightmost path, i.e., for* $i = n$.

Proof. The proof is by induction on the size of T. For $n = 1$ it is trivial. We assume that the statements hold for trees of the size smaller than n. First, we partition the set of paths: $\max_i \omega_T(i) = \max(\max_i \omega_{T_L}(i) + |T_L| - 1, \max_i \omega_{T_R}(i) + |T_R| - 1)$. Now, from the induction assumption and the definition of saving this value equals $\max(\mathsf{sav}(\mathsf{root}(T_L)), \mathsf{sav}(\mathsf{root}(T_R)))$ $= \mathsf{sav}(\mathsf{root}(T)) - |T| + 1$, which completes the first part of the proof. For the second path, observe, that the tree is sav-ordered, hence $\max(\mathsf{sav}(\mathsf{root}(T_L)), \mathsf{sav}(\mathsf{root}(T_R))) = \mathsf{sav}(\mathsf{root}(T_R))$. Finally, by induction assumption, we have $\mathsf{sav}(\mathsf{root}(T_R)) = \omega_T(n)$. □

For a tree S and $i \in [n]$, by S^i we denote the tree obtained from S as follows.

- Let $v_1, v_2 \dots v_k$ be nodes on the path from the root to the i-th leaf.



$dc(G, S_1) = 6$ $\qquad$ $dc(G, S_2) = 4$

Figure 4: Embeddings of a simple labeled gene tree $G = (a,(b,(c,(d,(e,(f,g))))))$ into species trees S_1 and S_2. The rightmost paths are colored in yellow. The number of extra lineages is shown near the corresponding edges. S_2 has simple labeling, therefore the right embedding is optimal. In particular, its rightmost path has no extra lineages.

- For $j = 1, 2, \dots, k$, if the left child of v_j is v_{j+1} then swap the subtrees of v_j.

Obviously, this transformation does not change the deep coalescence so $\widetilde{\mathsf{dc}}(G, S) = \widetilde{\mathsf{dc}}(G, S^i)$ for any i, G and S. Also, if S is sav-ordered then $S^n = S$. Now we can formulate our main theorem for the case of caterpillar gene trees.

Theorem 3. *Let* $\mathcal{C}_n$ *be a caterpillar gene tree, and* S *be a species tree. Assume that both are* sav*-ordered. If both have simple labeling, then* $\widetilde{\mathsf{dc}}(\mathcal{C}_n, S) = \mathsf{dc}(\mathcal{C}_n, S)$.

Proof. For simplicity, let $\overline{E}_n(S) = E(S) \setminus E_n(S)$. Note that the n-th leaf in a simple labeled $\mathcal{C}_n$ is the deepest node in $\mathcal{C}_n$ and $\Lambda_{\mathcal{C}_n}(n) = x_n$.[3] By $\widetilde{\mathsf{dc}}_i(\mathcal{C}_n, S)$ we denote the minimal $\mathsf{dc}(\mathcal{C}_n, S)$ in the set of all species trees S with the labeling satisfying $\Lambda_S(i) = x_n$. We split the proof into two parts. First, we show that $\widetilde{\mathsf{dc}}_i(\mathcal{C}_n, S)$ is determined by the simple labeling of S^i and equals $\sum_{e \in \overline{E}_n(S^i)} (|S^i(e)| - 1)$. Secondly, we prove that this sum is minimal if $i = n$.

Part I. Let $\Lambda_S(i) = x_n$. Clearly, $\widetilde{\mathsf{dc}}_i(\mathcal{C}_n, S) = \widetilde{\mathsf{dc}}_n(\mathcal{C}_n, S^i)$, i.e., we consider S^i with the n-th leaf labeled by x_n. Note, that every internal node of $\mathcal{C}_n$ maps to a node from the path $E_n(S^i)$ as x_n is the label of the n-th node from S^i and $\mathcal{C}_n$. Hence, if $e = \langle v, w \rangle$ is an edge from $\overline{E}_n(S^i)$, e is a lineage for every taxon (leaf) below v when embedding $\mathcal{C}_n$ into S^i. Thus, e is exactly $|S^i(e)|$ times a lineage, which gives $|S^i(e)| - 1$ extra lineages. We conclude that $\widetilde{\mathsf{dc}}_n(\mathcal{C}_n, S^i) \geq \sum_{e \in \overline{E}_n(S^i)} (|S^i(e)| - 1)$. Now, we show that this boundary is reached by the simple labeling of S^i. In such a case, for $j < n$, the lineages of the edge adjacent to the j-th leaf of $\mathcal{C}_n$ are disjoint with the rightmost path of S^i. Moreover, there is no extra

[3] Recall that x_n is the last taxon from the taxon set X.

lineage in $E_n(S^i)$ (see the right embedding in Fig. 4). This completes the proof of the first part.

Part II. Let $W(S) = \sum_{e \in S}(|S(e)| - 1)$. Note that $W(S) = W(S^i)$. It follows from the first part that $\widetilde{\mathsf{dc}}_n(\mathcal{C}_n, S^i) = \sum_{e \in \overline{E}_n(S^i)}(|S^i(e)| - 1) = W(S^i) - \sum_{e \in E_n}(|S^i(e)| - 1) = W(S) - \omega_n(S^i) = W(S) - \omega_i(S)$. Hence, we have $\widetilde{\mathsf{dc}}(\mathcal{C}_n, S) = \min_i \widetilde{\mathsf{dc}}_i(\mathcal{C}_n, S) = W(S) - \max_i \omega_i(S)$. Finally, by Lemma 1 we have that $\widetilde{\mathsf{dc}}(\mathcal{C}_n, S) = W(S) - \omega_n(S)$, i.e., when $i = n$ and $S^n = S$ is simply labeled. This completes the proof. □

Theorem 4. *If S is* sav*-ordered then*

$$\widetilde{\mathsf{dc}}(\mathcal{C}_n, S) = \sum_{e \in E(S) \setminus E_n(S)} |S(e)| - 1. \qquad (4)$$

Proof. Under the notation from the second part of the proof of Thm. 3 we have $\widetilde{\mathsf{dc}}(\mathcal{C}_n, S) = W(S) - \omega_n(S)$. The rest follows by expanding $W(S)$ and $\omega_n(S)$. □

The formula (4) allows us to compute the minimal deep coalescence cost in $O(n)$ time. Now we can compute easily the minimal cost for the complete balanced species tree when $n = 2^k$. We have $\widetilde{\mathsf{dc}}(\mathcal{C}_n, \mathcal{B}_n) = \sum_{i=1}^{k}(2^i - 1)(2^{k-i} - 1) = \sum_{i=1}^{k}(2^k - 2^i - 2^{k-i} + 1) = k2^k - (2^{k+1} - 2) - 2^k(1 - 2^{-k}) + k = 2^k(k - 2 - 1) + 2 + 1 + k = n(\log_2 n - 3) + \log_2 n + 3$.

It shows that $\widetilde{\mathsf{dc}}$ for the caterpillar and the complete binary tree is $\sim n \log_2 n$, which is similar to the results obtained in the previous section. Having this, one may also conjecture, that the maximal $\widetilde{\mathsf{dc}}$ for a caterpillar gene tree versus any species tree, both of the size n, is $\sim \frac{n}{2} \log_2 n$.

3.3 Algorithms for Gene-Species Matching

Here, we propose several heuristic algorithms for solving our problems. The algorithms, given the input consisting of two unlabeled trees of the same size, alter the ordering of nodes and infer labelings that approximate the minimal deep coalescence cost. Then, to compute the dc cost for such trees, we use the classical $O(n)$ algorithm based on LCA queries [3].

Algorithm 1: The simple algorithm

1: **Input:** Trees G and S of the same size.
2: **Output:** Approximation of $\widetilde{\mathsf{dc}}(G, S)$.
3: Order G by size and S by saving.
4: Add simple labelings to G and S.
5: **Return** $\mathsf{dc}(G, S)$.

Alg. 1 has a linear time and space complexity. Next, it follows from Thm. 1 and 3, that the simple algorithm is exact if one of the input trees is caterpillar as for caterpillar trees ordering by size and by saving are equivalent.

Algorithm 3: Extended greedy algorithm

1: **Input/Output:** see Alg. 2.
2: *Notation:* For a tree T, let $K(T)$ be the set of maximal nodes v, such that the left and the right subtree of v are isomorphic.
3: Order G by size and S by saving.
4: **return** $\min(d(G, S), \min_{g \in K(G), s \in K(S)} d(G^g, S^s))$, where G^g (and similarly S^s) is a tree obtained from G by swapping subtrees of v_j if, for each $j < k$, v_{j+1} is the left child of v_j, where $v_1, v_2, \ldots, v_k$ is the path from the root of G to g.
5: **Function** $d(G, S)$: all lines ≥ 4 from Alg. 2

Although the simple algorithm fits our theorems perfectly, one could find even small counterexample when the output cost is not optimal. Therefore, we propose another approach (see Alg. 2), in which we first try to match *cherries*, which are nodes with precisely two leaves beneath. Empirical evaluation shows, see Fig. 5 that the greedy algorithm performs better than Alg. 1 in terms of the returned cost. Alg. 2 has a quadratic time complexity. It is also more difficult to find a counterexample which does not give the lowest cost.

Extending algorithms. To further improve the performance of our algorithms we propose to apply different kinds of orderings in some nodes of the input trees. The details how to extend Alg. 2 are depicted in Alg. 3. Analogously, we extend the simple algorithm. Both extended algorithms are never worse than the original ones, and we still have the exact solution for caterpillar trees. For the other trees, extended algorithms tend to perform better, which is summarized in Fig. 5. As the set of nodes $K(T)$ in Alg. 3 can be computed by using an $O(n \log n)$ the solution proposed by Campbell et al. [19], the time complexity of the extended greedy algorithm is $O(n^4)$, while the extended variant of the simple algorithm requires $O(n^3)$ time.

The evaluation of all proposed algorithms is depicted in Fig. 5.

4 Experimental Results

We have performed two computational experiments on empirical and simulated datasets. In the first experiment, we present a comparative study of the reconstruction algorithms, while in the second, we tested the quality of labeling inference.

Experiment I. To verify which algorithm yields

Algorithm 2: The greedy algorithm

1: **Input:** Trees G and S of the same size. **Output:** Approximation of $\widetilde{\mathsf{dc}}(G, S)$.
2: *Notation:* For a tree T and a set of nodes $Z \subseteq V(T)$ and $i \in \{1, 2, \ldots, |Z|\}$, by $Z[i]$ we denote i-th node from Z when T is traversed in post-order. By I_T we denote the set of internal nodes of a tree T.
3: Order G by size and S by saving.
4: Add the simple labeling to G. Let $i := j := 1$. Initialize sets $M_G := M_S := \emptyset$.
5: $F := C_G(\mathsf{root}(G))$ - the set of unmapped leaves from G; $U := C_S(\mathsf{root}(S))$ - the set of unlabeled leaves in S.
6: **While** $j \leq n - 1$
7: $A := |F \cap C_G(I_G[i])|$ and $B := |U \cap C_G(I_S[j])|$
8: **If** $|A| = |B|$ **Then** $\mathsf{map}(A, B)$; i+=1; j+=1;
9: **Else If** $|A \cup M_G| = |B \cup M_S|$ **Then** $\mathsf{map}(A \cup M_G, B \cup M_S)$; $M_G := M_S := \emptyset$; i+=1; j+=1;
10: **Else If** $|A \cup M_G| = |B|$ **Then** $\mathsf{map}(A \cup M_G, B)$; $M_G := \emptyset$; i+=1; j+=1;
11: **Else If** $|A| = |B \cup M_S|$ **Then** $\mathsf{map}(A, B \cup M_S)$; $M_S := \emptyset$; i+=1; j+=1;
12: **Else If** $|A| < |B|$ **Then** $M_G := M_G \cup A$; i+=1;
13: **Else** $M_S := M_S \cup B$; j+=1;
14: **return** $\mathsf{dc}(G, S)$.
15: **Function** $\mathsf{map}(P, Q)$:
16: **For** $k = 1, 2, \ldots, |P|$, set the label of $Q[k]$ to be the label of $P[k]$.
17: $F := F \setminus P$, $U := U \setminus Q$.

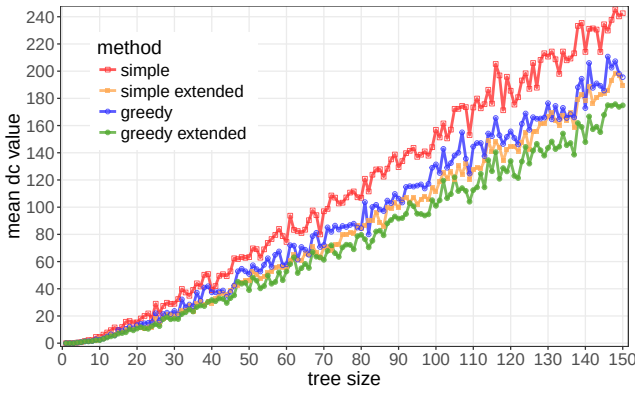


Figure 5: Averaged minimal deep coalescence, computed by all four heuristics.

the lowest cost, we generated random trees from the Yule model [20]. For each tree size between 1 and 150 we generated 7 random tree pairs. Then, we computed the approximation of the minimal cost by using our four algorithms. The result, averaged over sizes of trees, is depicted 5. We observe that the greedy extended algorithm is the best performing among all our algorithms.

Experiment II. In practice, we often have some partial information on the labeling of leaves. Therefore, we introduce a more practical variant of our problems:

Given a gene tree G with a partial labeling, i.e., some leaves of G are unlabeled, and a species tree S. Find the total labeling for G that minimize the deep coalescence cost $\mathsf{dc}(G, S)$.

The greedy algorithm can be easily modified to solve the above problem. In line 5 of Alg. 2, it is sufficient to remove labeled leaves from F and used taxons being labels from U.

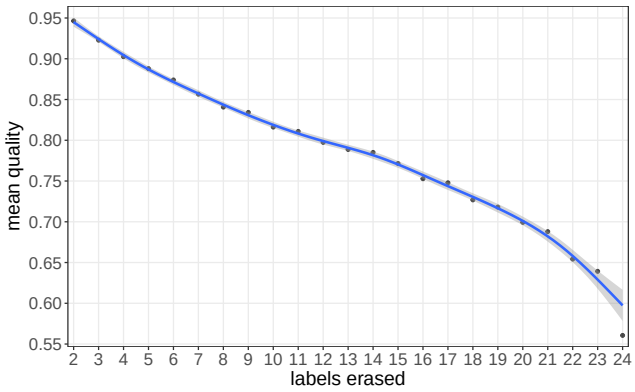


Figure 6: Averaged quality, computed by the modified greedy algorithm on the trees from *TreeFam* dataset [21].

The test was performed on TreeFam dataset, which consists of 1274 curated gene family trees from TreeFam v7.0 [21] spanning 25 mostly animal species. The species tree was based on the NCBI taxonomy. First, we extracted 295 gene trees with bijectively labeled leaves. Next, for each such a gene tree G, we contracted a species tree to the taxons present in G. Hence, we obtained 295 pairs of bijectively labeled trees, with the average size of 17.66 taxons. Finally, for every pair of trees $\langle G, S \rangle$, and for each $i = \{2, 3, \ldots, |G|\}$, we removed labels of i random leaves from G and then applied the modified greedy algorithm to infer total labeling. The quality of a reconstruction is determined by the number of properly reconstructed

labels divided by i. The experiment was repeated 10 times. The result, averaged over i, is depicted in Fig. 6.

5 Conclusion

In this article, we gave a closer look at an open question of the minimal deep coalescence cost for the fixed tree topologies and bijective labelings. The particular cases that we have solved provide a better understanding of the properties of the deep coalescence cost. While the complexity of our problems remains open, the methods presented seem to be a good starting point to the solve the problems in the general case which we plan to study in future. Also, we plan to test our solutions on more complex empirical and simulated datasets, including simulations with more realistic models.

Acknowledgements

The support was provided by NCN grant #2015/19/B/ST6/00726.

References

[1] W. P. Maddison. "Gene Trees in Species Trees". In: *Syst Biol* 46 (1997), pp. 523–536.

[2] R. D. M. Page and E. C. Holmes. *Molecular evolution: a phylogenetic approach*. Blackwell Science, 1998.

[3] L. Zhang. "From Gene Trees to Species Trees II: Species Tree Inference by Minimizing Deep Coalescence Events". In: *IEEE/ACM TCBB* 8 (2011), pp. 1685–1691.

[4] B. C. Carstens and L. L. Knowles. "Estimating species phylogeny from gene-tree probabilities despite incomplete lineage sorting: an example from Melanoplus grasshoppers". In: *Syst Biol* 56.3 (2007), 400–11.

[5] W. Jennings and S. Edwards. "Speciational history of Australian grass finches (Poephila) inferred from thirty gene trees". In: *Evolution* 59.9 (2005), 2033–47.

[6] C. Than and L. Nakhleh. "Species tree inference by minimizing deep coalescences". In: *PLoS Comput Biol* 5.9 (2009), e1000501.

[7] C. V. Than and N. A. Rosenberg. "Consistency Properties of Species Tree Inference by Minimizing Deep Coalescences". In: *J Comput Biol* 18.1 (2011), pp. 1–15.

[8] Y.-C. Wu, M. D. Rasmussen, M. S. Bansal, and M. Kellis. "Most parsimonious reconciliation in the presence of gene duplication, loss, and deep coalescence using labeled coalescent trees". In: *Genome research* 24.3 (2014), pp. 475–486.

[9] R. Chaudhary, J. G. Burleigh, and O. Eulenstein. "Efficient error correction algorithms for gene tree reconciliation based on duplication, duplication and loss, and deep coalescence". In: *BMC bioinformatics* 13.10 (2012), S11.

[10] P. Górecki and O. Eulenstein. "Deep coalescence Reconciliation with Unrooted Gene Trees: Linear Time Algorithms". In: *LNCS* 7434 (2012), pp. 531–542.

[11] P. Górecki, O. Eulenstein, and J. Tiuryn. "Unrooted Tree Reconciliation: A Unified Approach". In: *IEEE/ACM TCBB* 10.2 (2013), pp. 552–536.

[12] P. Górecki and O. Eulenstein. "Maximizing Deep Coalescence Cost". In: *IEEE/ACM TCBB* 11.1 (2014), pp. 231–242.

[13] C. V. Than and N. A. Rosenberg. "Mathematical properties of the deep coalescence cost". In: *IEEE/ACM TCBB* 10.1 (2013), pp. 61–72.

[14] P. Górecki and O. Eulenstein. "Gene Tree Diameter for Deep Coalescence". In: *IEEE/ACM TCBB* 1 (2015), pp. 155–165.

[15] A. Mykowiecka, P. Szczesny, and P. Gorecki. "Inferring gene-species assignments in the presence of horizontal gene transfer". In: *IEEE/ACM TCBB* (2017).

[16] A. Betkier, P. Szczęsny, and P. Górecki. "Fast Algorithms for Inferring Gene-Species Associations". In: *International Symposium on Bioinformatics Research and Applications*. Springer. 2015, pp. 36–47.

[17] L. Zhang and Y. Cui. "An Efficient Method for DNA-Based Species Assignment via Gene Tree and Species Tree Reconciliation". In: *WABI*. Springer. 2010, pp. 300–311.

[18] P. Górecki and J. Tiuryn. "DLS-trees: A model of evolutionary scenarios". In: *Theor Comput Sci* 359.1-3 (2006), pp. 378–399.

[19] D. M. Campbell and D. Radford. "Tree isomorphism algorithms: speed vs. clarity". In: *Mathematics Magazine* 64.4 (1991), pp. 252–261.

[20] G. U. Yule. "A mathematical theory of evolution, based on the conclusions of Dr. JC Willis, FRS". In: *Philosophical transactions of the Royal Society of London. Series B, containing papers of a biological character* 213 (1925), pp. 21–87.

[21] J. Ruan et al. "TreeFam: 2008 Update". In: *Nucleic Acids Res* 36 (2008), pp. D735–40.

State-of-the-art Genomic Data Compression Technology

Dunling Li[1] , Lin-Ching Chang[2]
[1]BTS Software Solutions, Columbia, Maryland, USA
[2]Dept. of Electrical Engineering and Computer Science, The Catholic University of America, Washington, District of Columbia, USA
dunling.li@bts-s2.com, changl@cua.edu

Abstract — The demand for efficient genomic data storage and distribution has increased substantially as advanced high-throughput sequencing has dramatically reduced costs and processing time required for genomic data collection. However, these advances generate a new challenge with data storage, transmission, and manipulation. To wit, the sizes of genomic databases are often very huge; for example, the genomic data in Cancer Genome Atlas database is over 2.5 petabytes (2.5 million gigabytes). Data compression would be the key to meet the technical and cost challenges of genomic data storage, transmission, and access for analysis. Recently, there has been growing interest in developing high-performance compression tools designed specifically for genomic data. This paper provides an overview of state-of-the-art genomic data compression technology, the commonly used tools, and their performance.

Keywords: Genomic data compression, Sequencing Data Compression, Reads Compression, Genomic Compression Tools, Compression Performance.

I. INTRODUCTION

As the next-generation sequencing (NGS), also known as high-throughput sequencing (HTS), technology advances, sequencing genomes is now much faster and cheaper. It makes larger and larger whole-genome projects like 1000 Genomes feasible [1]. For example, the cost of sequencing a whole human genome has dropped from $20 million in 2004 to $1,000 in 2015 [2]. Sequencing the first human genome took 13 years to complete [3] while we only need an hour to complete one human genome in 2017 [4]. The data generated during the first 6 months by the 1000 Genomes Project, an international research effort to establish the most detailed catalogue of human genetic variation from 2008 to 2013 [1], has exceeded the sequence data accumulated during 21 years in the NCBI GenBank database [5].

The significant cost and time reduction has led to massive amounts of genomic data being generated at ever increasing rates. Currently, sequencing data is doubling approximately every seven months. At such a fast rate, more than an exabyte (i.e. 10^{18} bytes) of genomic data will be generated every year, and the total amount of sequencing data will reach a zettabyte (i.e., 10^{21} bytes) by 2025 [6].

Due to the unique nature of genomic data, whose samples may not be available in the future for re-sequencing from organisms and/or ever-changing ecosystems, data must be stored and preserved without information loss. The rapid increase in genomic data generation and resulting huge omnipresent HTS datasets lead to great challenges in storing and distributing massive amount of genomic data. Obviously, data compression is the technology to solve these challenges [2, 7-9]. Genomic data compression has attracted many in the field to try and develop high performance genome-specific methods [2, 7, 9].

Moreover, current trends in HTS data generation indicate that the storage, transmission, and bandwidth costs will soon surpass the costs of sequencing. These issues will become the main bottleneck in genomics research and in the applications of HTS data to precision medicine. To meet this growing demand of HTS data storage and distribution, the Moving Picture Expert Group (MPEG) and HTS industry formed the MPEG HTS Compression Working Group (MPEG-HTS-CWG) in 2015 [10]. The ultimate goal of this group is to design and specify genomic data compression and transport technology by means of an open standard and interoperability among systems. This paper will provide a concise overview on genomic data, state-of-the-art compression methods including both lossless and lossy approaches, existing compression tools, and ongoing research and development.

II. GENOMIC DATA OVERVIEW

Genomic data refers to the genome and DNA data of an organism. It is used in bioinformatics for collecting, storing, and processing the genomes of living things. Genomic information is presented as genome sequencing, a long sequence of DNA nucleotides with specific orders, using NGS or whole genomic sequencing (WGS). NGS technologies are not able to provide the whole genome sequence, but produce a collection of millions of small fragments. WGS is the process of determining the complete DNA sequencing of an organism's entire genome at one time by many fragments acquired using NGS. Therefore, most of the genomic data being stored and analyzed to date is comprised of sequencing data produced by NGS technologies.

The fragments NGS produces are called 'reads.' As the reads may have a wrong readout of the actual sequencing signal, the raw reads include both the reads and their corresponding quality scores, indicating the probability of a given base that is called incorrectly by the sequencer [2, 11]. The number of raw reads in the sequencing data depends on the coverage, which the expected number of times a specific nucleotide of the genome is sequenced. For most of NGS technologies, the read length is of a few hundred base-pairs, which is much smaller than a full genome; for comparison, it would be about 3 billion base-pairs for a human genome.

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

Each base-pair is a combination of four nucleobases: Cytosine (C), Guanine (G), Adenine (A) or Thymine (T). Reads are typically stored in FASTQ format, a text-based format for storing both the sequence letters and quality score which are each encoded with a single ASCII character for brevity [12].

After generating raw sequencing data, NGS mechanisms typically apply mapping algorithms [13] to align the reads to a reference sequence to determine the corresponding location in the reference for each read. The alignment information along with original reads and their quality scores are stored in the standard Sequence Alignment Map (SAM) file format or Binary Alignment Map (BAM) - a compressed format of SAM files.

NGS also generates other files which contain additional information between the original genome and the one used as a reference. To illustrate the size of these files, a human genome containing 3 billion of base-pairs is about 3GB of data if the base-pairs are stored in FASTQ format. Typical NGS of a human genome with 200x coverage will generate around 6 billion reads, assuming the read length of 100bp. The resulting FASTQ file and SAM file will be about 1.5TB and 3TB respectively [2].

III. Genomic Data Compression Methods

Although standard data compression tools like zip and rar have been used to compress genomic data, they do not exploit the particularities of the data itself and yield a relatively low compression gain. As genomic sequences often contain repetitive contents and many sequences exhibit high levels of similarity, specialized new lossless compression methods have been developed in past few years [15-18]. Other existing lossless compression algorithms can be categorized into Naïve Bit encoding [19, 20], Dictionary-based [21, 22], Statistical-based [23, 24], and Reference-based encodings [2, 25, 26].

Naïve Bit encoding methods exploit fixed-length encodings of two or more symbols in a single byte. Such encoding methods were designed for raw DNA sequencing and yield a range of compression ratios from 2:1 to 6:1 [9, 15-18]. Of note, Naïve Bit encoding methods may not be suitable to other types of genomic data such as quality scores due to the large size of their symbol sets.

Dictionary-based compression methods replace repeated substrings by references, which is a set of previously seen or predefined common strings and can be built at runtime. Lampel-Ziv-based compression algorithms, such as zip or bpzip2, are prominent examples of dictionary-based algorithms [27]. Usually the dictionary and reference indices are further encoded by entropy coding methods like arithmetic coding or integer coding schemes like Golomb codes [28]. Such approachs can be applied to any type of genomic data. The state-of-the-art dictionary-based genomic compression methods achieve a compression ratio in the range of 4:1 to 6:1 depending on the frequency of repeats in the specific genomes being compressed [9, 29, 30].

Statistical-based encoding methods derive a probabilistic model from the input data. Based on the partial matches of subsets of the input, the model predicts the next symbols in the sequence. Statistical encoding can be used in all types of genomic data. The state-of-the-art statistical based methods are able to reach compression ratios in the range of 4:1 to 8:1 [9, 24, 31-34] depends on the prediction accuracy.

Referential approaches replace long substrings of the to-be-compressed input with references to another string. They require that the references are available for both the encoding and decoding process as they are not part of input data. Reference-based genomic compression algorithms are designed to utilize the extremely high level similarity of all re-sequenced genomes from the same species. Such methods can reach very high compression ratios (approaching 400:1) if the reference sequences well represent the to-be-encoded sequences [9, 35, 36].

The aforementioned compression methods are all **lossless** or **reversible**, i.e. the decompressed data is identical to the original input data. To compress genomic data even further, **lossy** compression methods have been proposed for the quality score portion of the NGS data. However, quality scores contain noise induced during sequencing and are harder to compress than the corresponding reads; evidence shows that quality scores can occupy over 70% of compressed genomic data files [2, 37].

Quality scores generated by base-calling algorithms in HTS are not strictly part of the DNA sequences. The scores are supplementary information used in a heuristic (somewhat inaccurate) manner when downstream applications operate on the reads. Observation of this behavior shows that minor perturbation of these values does not affect downstream analysis and manipulation tools. Thus, lossy compression is a viable technique for compression of quality scores [2]. Various lossy compression algorithms for genomic data have been proposed [38-41]; their performance is usually shown in rate-distortion representations. In some examples, over 10:1 compression on quality scores had no corresponding negative impact on application results [2].

IV. Genomic Data Lossless compression Tools and Their Performance

Many HTS data compression tools for FASTQ and SAM files are available in both commercial markets and open source. The MPEG-HTS-CWG has done a comprehensive evaluation on most of the available open-source compression tools for HTS data, which include both industrial-scale tools as well as research-oriented prototypes [10, 42]. In Figure 1, we summarized the state-of-the-art compression tools that are currently available to compress FASTQ and/or SAM files. Compression tools showed in the red box, i.e., pigz (parallel gzip), pbzip2 (parallel bzip) and Quip, can be used for both FASTQ and SAM files. Pigz and pbzip2 are dictionary-based

generic lossless compression tools and commonly used for FASTQ compression designed to take advantage of multi-processor/multi-core systems.

Specialized FASTQ compressors initially perform a form of transformation such as read-identifier tokenization or 2-bit nucleotide encoding, followed by statistical modeling and entropy coding. Examples of such approaches are DSRC2, FQC, Fqzcomp and Fastqz, Slimfastq, and LFQC as showed in Figure 1. Because the read order within a FASTQ file is arbitrary and most of the underlying genome is repetitive, the other approach is re-ordering the reads in a manner that brings similar reads together to boost compression ratios. Tools like SCALCE, Orcom, Mince, and BEETL use this approach as a preprocessing step to improve compression performance.

Another approach is replacing each read with a pointer to the underlying reference genome, assuming such a reference genome is available at both encoding and decoding sides. FQZip is an example of this approach. Where a reference genome is not available, the same technique can be used after *de novo* assembly. Many *de novo* techniques exist though the most common is the assembly of contigs. References to newly assembled contigs or de Bruijn paths can then be substituted for sequences. The primary tools that use assembly for data compression are Quip, Leon, k-Path, and KIC. However, both sequence mapping and assembly are computationally intensive tasks. Tools like Orcom, BEETL, Mince, and k-Path are designed for FASTQ reads only files. The tools DRSC2, Fqzcomp, Fastqz, FQC, CALCE, FQZip, Leon, KIC, etc (in purple box) are used for FASTQ full files and reads only files.

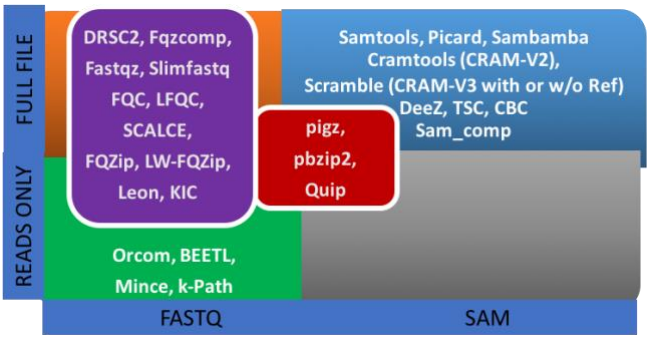


Figure 1. Compression Tools for FASTQ and SAM files

Most SAM files are compressed and stored in the BAM format. The compression tools for BAM manipulation are Samtools, Picard, and Sambamba. All BAM-based tools support arbitrary ordering of the reads and do not require a reference during compression or decompression. None of them treat various streams in a BAM file differently. The reference-based compression for SAM files is CRAM which separates different fields in the reads and applies different compression techniques on each. CRAM is implemented in Cramtools, and Scramble has recently been incorporated into Samtools and Picard.

In both BAM and CRAM formats, reads covering the same sequence variant are encoded independently. To reduce the redundancy among the same variants, DeeZ implicitly assembles the underlying donor genome to encode these variants only once. CBC or TSC follows a similar path, encoding variants only one time. All of these tools treat each SAM field independently and apply a variety of compression techniques to each field. Finally, Quip and sam_comp employ highly optimized statistical models for various SAM fields; they are among the best performing tools in terms of pure compression ratio [42].

To compare the performance of these compression tools, the MPEG-HTS-CWG compiled a broad HTS dataset with a wide spectrum of characteristics to ensure statistically meaningful results from the compression performance evaluation. The dataset includes FASTQ and SAM files with both deep and shallow coverage; fixed-length and variable length reads obtained by sequencing technology from leading manufacturers (Illumina, Pacific Biosciences, etc.); genomic data from various organisms (Homo sapiens, bacteria, plants, etc.); and several sample types (metagenomics, cancer cell lines, etc.). We summarize MPEG-HTS-CWG test dataset in Table 1 and Table 2. It contains 7 FASTQ files and 8 SAM files respectively. The total data size is about 4 TB [10, 42].

Table 1 Selected FASTQ Files

#	Samples	Full-Size (MB)	Reads-Size (MB)	Cove-rage (MB)	Organism	Techs
1	SRR554 369	650	165	25	Pseudomona s aeruginosa	Illumina GAIIx
2	SRR327 342	3881	947	80	Saccharomyc es cerevisiae	Illumina GAII
3	MH0001 .081026	1880	512	N/A	Homo sapiens gut	Illumina GA
4	SRR128 4073	1309	649	140	Escherichia coli	PaciBio
5	649SRR 870667	22944	7463	20	Theobroma cacao	Illumina GAIIx
6	ERR174 310	53869	20966	7	Homo Sapiens	HiSeq
7	ERP001 775	2717029	1059387	120	Homo Sapiens	HiSeq

Table 2 Selected SAM Files

#	Samples	Full-Size (MB)	Coverage (MB)	Organism	Techs
1	DH10B	5579	420	E. Coli	MiSeq
2	9827.2.4 9	21059	2	Homo Sapiens	HiSeq
3	sample- 2-1	5924	0.6	Homo Sapiens	Ion Torrent
4	K562.LI D8465	75915	6	Homo Sapiens	RNASeq
5	dm3	30081	75	Drosophila melanogaster	Pacbio
6	NA1287 8.PB	126545	15	Homo Sapiens	Pacbio
7	HCC195 4	427028	30	Homo Sapiens	Cancer cell
8	NA1287 8.S1	589038	50	Homo Sapiens	HiSeq

MPEG-HTS-CWG evaluated the performance of genomic data compression tools in term of compression ratio, relative encoding and decoding times, memory usage, and parallelization capabilities [10]. In this paper, we focus on compression ratios and speed; we present MPEG-HTS-CWG test results of the top performing tools, while more details can be found in references [10] and [42]. Table 3, 4 and 5 show the corresponding compression ratios of HTS compression tools on FASTQ full files, FASTQ reads only, and SAM files. A FASTQ full file includes reads and quality scores. FASTQ files are typically compressed by general purpose pigz and pbzip2 tools while SAM files are compressed by Samtools.

Note that pbzip2 performs significantly better than pigz in all test samples in Table 3, 4, and 5. Neither pbzip2 nor Samtools have the best performance in any cases among the HTS specific compression tools. The tools that yield the highest compression ratios for each test file are marked in red for easy reading. No one tool out-performs on all files.

Table 3 Compression Ratios on FASTQ Full Files

Sample	1	2	3	4	5	6	7
DSRC2	6.2	5.8	4.4	N/A	4.8	4.1	N/A
Fqzcomp	7.3	6.9	4.9	N/A	5.7	4.8	N/A
lfqc	9.4	7.9	6.7	3.2	9.5	N/A	N/A
pbzip2	5.2	4.7	3.5	2.8	4.1	3.6	11.2
pigz	4.1	3.8	2.8	2.4	3.3	2.9	8.9
Quip	7.3	7.2	5.1	3.1	5.9	4.8	14.8
SCALCE	8.6	8.0	4.6	3.1	6.2	5.0	16.9
slimfastq	6.9	7.7	5.4	N/A	5.4	4.9	15.3

Table 4 Compression Ratios on FASTQ Reads Only Files

Sample	1	2	3	4	5	6	7
DSRC2	4.0	3.7	4.0	N/A	4.0	4.0	N/A
Fqzcomp	4.5	4.7	4.3	N/A	4.8	4.5	N/A
lfqc	9.7	7.3	5.0	4.2	N/A	N/A	N/A
pbzip2	3.8	3.8	3.7	3.7	4.0	3.8	11.0
pigz	3.4	3.4	3.4	3.5	3.5	3.5	10.1
Quip	4.5	5.2	4.5	4.1	5.1	4.6	13.3
SCALCE	9.7	13.9	7.2	4.0	7.5	6.9	37.2
slimfastq	5.5	6.4	4.9	N/A	5.3	4.7	13.6
BEETL	7.2	8.1	4.5	N/A	6.2	5.4	N/A
k-Path	11.8	21.0	8.3	N/A	11.3	10.0	N/A
Mince	16.5	25.6	10.2	N/A	10.9	10.7	N/A
Orcom	15	26.3	10.0	N/A	9.0	11.7	153.1

Among the tested FASTQ tools, the best compression ratios are offered by tools that reorder reads. Sequence-mapping and assembly-based tools like lfqc may also provide good compression ratios, but they are often slow with high memory costs. Table 3 and 4 show lfqc and SCALCE are the best tools for FASTQ full files while Orcom and Mince yield the highest compression ratios for FASTQ reads-only files. Note that the majority of the available tools are optimized for Illumina-style short, fixed-length reads; many tools are not capable of compressing long variable-length read collections such as data from Pacific Biosciences (PacBio).

For SAM files, it is possible to achieve better compression ratios than those achieved by Samtools. However, unlike Samtools, they do not provide random-access capability. Among the available tools, only BAM and CRAM-family tools, DeeZ and TSC are able to conduct random-access. Table 5 shows DeeZ and Quip with reference reach the best compression ratios. The integrated tool sam_comp at the last row of Table 5 yields the best compression ratios. The integrated solution chooses the specific approach which out performs on input data type with respect to the performance measure, i.e. compression ratio in Table 5.

Table 5 Compression Ratios on SAM Files

Sample	1	2	3	4	5	6	7	8
Cram2	5.2	5.6	5.1	7.3	4.0	3.3	4.4	N/A
Cram3	6.5	6.4	5.8	8.2	4.6	3.7	5.2	8.8
Cram3NR*	6.2	5.0	5.3	7.7	2.8	2.8	4.9	8.1
DeeZ	7.6	7.7	6.5	10.5	4.6	3.7	5.7	11.0
pbzip2	5.2	4.0	5.3	7.4	3.1	2.9	4.3	6.6
picard	3.9	3.2	4.0	5.5	2.3	2.2	3.2	N/A
Pigz	4.2	3.5	4.3	5.9	2.4	2.4	3.6	5.2
Quip	5.1	4.8	4.8	6.8	3.3	3.0	4.3	6.1
Quip_Ref	6.9	N/A	N/A	8.7	4.7	N/A	N/A	9.1
Samtools	4.0	3.2	4.0	5.5	2.3	2.2	3.2	4.8
sam_comp	8.0	7.9	6.7	10.8	3.6	3.9	10.0	11.1

Cram3NR*: CRAM Version 3 without Reference

In general, there is trade-off among compression ratio, compression (encoding) and decompression (decoding) speed, memory cost, etc. Table 6 and 7 show the encoding and decoding speed measures of selected FASTQ tools. The processing speed is usually measured as a ratio of data size over processing time, e.g. MB/sec, and depended on the power of processor. For the purpose of comparing different tools, the speed measures are defined as the ratio of the processing times of a given tool over a reference tool. Therefore, the smaller the speed measure, the faster the actual speed and the lower the computation cost.

Table 6 FASTQ Encoding Speed Measures

Sample	1	2	3	4	5	6	7
DSRC2	0.22	0.26	0.24	N/A	0.21	0.2	N/A
Fqzcomp	0.34	0.37	0.41	N/A	0.33	0.32	N/A
lfqc	18.53	18.5	21.1	18.03	14.5	N/A	N/A
pbzip2	1.19	1.45	1.29	0.74	0.99	0.81	0.21
pigz	1	1	1	1	1	1	1
Quip	0.5	0.53	0.47	0.36	0.48	0.46	0.38
SCALCE	0.77	0.63	0.8	0.67	0.6	0.59	0.57
slimfastq	0.55	0.47	0.54	N/A	0.51	0.47	0.49

Table 7 FASTQ Decoding Speed Measures

Sample	1	2	3	4	5	6	7
DSRC2	2.11	3.09	1.91	N/A	1.39	1.22	N/A
Fqzcomp	N/A	7.54	N/A	N/A	N/A	3.29	N/A
lfqc	315	310	339.9	386	N/A	N/A	N/A
pbzip2	5.97	6.85	6.35	6.99	3.61	2.83	1.23
pigz	1	1	1	1	1	1	1
Quip	10.7	11.5	11.37	10.6	5.57	5.22	4.64
SCALCE	9.05	8.23	12.17	9.78	4.89	4.57	1.94
slimfastq	11.46	9.55	11.32	N/A	5.8	4.76	5.94

Table 6 shows the encoding speed measures, which are calculated using the corresponding encoding time divided by pigz encoding time. Similarly, the decoding speed measures in Table 7 are computed using the corresponding decoding time divided by pigz decoding times. The fastest tools are again marked in red. DSRC2 and pbzip2 are the fastest FASTQ encoding tools while pigz and DSRC2 are fastest

decoding ones. Comparing with Table 3, we can see that the best FASTQ tools, lfqc and SCALCE, marked in green in Table 6 and 7, are the costliest ones in terms of encoding and decoding times.

Table 8 SAM Encoding Speed Measures

Sample	1	2	3	4	5	6	7	8
Cram2	0.93	1.42	2.12	1.7	0.93	1.01	1.28	N/A
Cram3	0.23	0.29	0.62	0.38	0.14	0.31	0.27	0.1
Cram3NR	0.29	1.18	0.45	0.43	0.21	0.46	0.37	0.1
DeeZ	0.91	1.23	3.49	2.01	0.71	1.22	1.66	0.41
pbzip2	1.65	1.93	4.04	3.57	0.72	0.94	1.62	0.46
picard	1.42	1.04	1.82	1.48	0.74	0.55	1.18	N/A
pigz	0.77	1.55	1.48	1.06	1.39	1.37	1.4	0.13
Quip	10.7	7.81	4.43	8.27	7.52	9.87	9.05	2.18
Quip_Ref	10.1	N/A	N/A	8.2	7.19	N/A	N/A	2.2
Samtools	1	1	1	1	1	1	1	1
Sam_comp	0.68	0.76	1.2	0.71	0.51	0.59	0.62	0.37

Table 9 SAM Decoding Speed Measures

Sample	1	2	3	4	5	6	7	8
Cram2	1.71	1.67	4.93	2	2.05	2.39	1.5	N/A
Cram3	0.76	0.66	1.58	0.67	0.58	0.84	0.71	0.5
Cram3NR	0.74	0.63	1.06	0.78	1.14	1.54	0.79	0.47
DeeZ	10.1	5.6	9.86	7.91	4.86	6.67	6.39	1.9
pbzip2	3.16	3.39	3.72	2.46	2.93	3.94	3.23	0.59
picard	2.76	1.52	2.1	2.44	1.09	1	1.91	N/A
pigz	0.63	0.82	0.49	0.7	0.79	0.7	0.91	0.6
Quip	10.7	7.81	4.43	8.27	7.52	9.87	9.05	2.18
Quip_Ref	10.1	N/A	N/A	8.02	7.19	N/A	N/A	22
Samtools	1	1	1	1	1	1	1	1
Sam_comp	3.36	2.95	6.54	3.56	5.49	5.42	3.25	2

Similarly, Table 8 and 9 show the comparison results of the encoding and decoding speed measures of SAM tools. The encoding speed measures in Table 8 are the ratios of encoding time of different tools divided by Samtools' encoding time. The values in Table 9 are the decoding time of different tools divided by Samtools' decoding time. Cram3 and Cram3NR are the fastest encoding tools while pigz, cram3 and CramNR are the fastest decoding tools for SAM files. Comparing with Table 5, the tools that yield the highest compression ratios, DeeZ and Quip with reference, are marked in green in Table 8 and 9. Their encoding speeds lay in the middle compared with other SAM tools, but their decoding speeds are among the slowest. The integrated solution sam_comp yields the highest compression ratios in Table 3; its encoding and decoding speeds are in reasonably fast ranges in Table 8 and 9. Therefore, the integrated solution provides the best overall outcomes in term of compression ratio and speed.

V. Conclusion

As NGS technology continually advances, it is estimated that genomic data will be doubled every seven months. Data compression would be a key technology for efficient genomic data storage and distribution. In this paper, we have provided an overview of the state-of-the-art compression techniques for genomic data compression and summarized the performance evaluation results of widely used genomic data compression tools using the MPEG HTS dataset.

Genomic data mainly contains HTS datasets, which are stored in FASTQ or SAM formats. Their sizes and contents vary significantly based on the applications. The MPEG-HTS-CWG's evaluation results show that HTS data-specific compression tools achieve higher compression ratios than generic lossless compression tools, but no method performs the best in all test cases. The compression ratios of these state-of-the-art tools depend on the datasets and are typically perform below 10:1. Research to develop new methods to compress genomic data more efficiently and effectively is needed in the future of genomic data processing.

There are trade-offs between compression ratios and their corresponding encoding/decoding speeds. Integrated solutions are able to provide better performance in terms of compression ratios and provide reasonable encoding and decoding speed.

Various lossy compression methods have also been developed for the quality scores of HTS data, and research shows that over 10:1 compression on quality scores has no negative impact on application results. The MPEG-HTS-CSG is currently developing a standard format for compressed files to integrate the best features of the tools and formats. In our view, new algorithms to compress genome data producing both a high compression ratio and high quality of decompressed data in the future will be needed. The integrated solution would consist a dynamic combination of lossless and lossy methods targeting the original reads and their quality scores respectively.

VI. References

[1] https://en.wikipedia.org/wiki/1000_Genomes_Project

[2] Ochoa I. Genomic data compression and processing: theory, models, algorithms and experiments, PhD dissertation, Stanford University, 2016.

[3] https://www.livescience.com/28708-human-genome-project-anniversary.html

[4] http://www.sandiegouniontribune.com/business/biotech/sd-me-illumina-novaseq-20170109-story.html

[5] Re C, Ro A, Re A. Will computers crash genomics? *Science*, 5:1190, 2010.

[6] Stephens ZD, et al. Big data: Astronomical or genomical? *PLoS Biol*, 13(7):e1002195, 2015.

[7] https://en.wikipedia.org/wiki/Compression_of_Genomic_Re-Sequencing_Data

[8] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3868316/

[9] Wandelt S, Bux M, Leser U. Trends in Genome Compression. Current Bioinformatics. 9. doi:10.2174/157489360966140516010143.

[10] Numanagić I, Bonfield JK, Hach F,Vges J, Ostermann J, Alberti C, Mattavelli M, Sahinalp SC. Comparison of high-throughput sequencing data compression tools, Nature Methods, 13, 1005–1008 (2016). doi:10.1038/nmeth.4037

[11] Buermans HPJ, Dunnen JTD. Next generation sequencing technology: advances and applications. *Biochimica et Biophysica Acta (BBA)-Molecular Basis of Disease*, 1842(10):1932–1941, 2014.

[12] https://www.decodedscience.org/comparing-genetic-code-dna-binary-code/55476

[13] Li H, et al. The sequence alignment/map format and samtools. *Bioinformatics*, 25(16):2078–2079, 2009.
[14] Cock PJ, Fields CJ, Goto N, Heuer ML, Rice PM. The sanger fastq file format for sequences with quality scores, and the solexa/illumina fastq variants. *Nucleic acids research*, 38(6):1767–1771, 2010.
[15] Vey G. Differential direct coding: a compression algorithm for nucleotide sequence data. The Journal of Biological Databases and Curation, 2009.
[16] Rajeswari PR, Apparo A, Kumar VK. Genbit compress tool(gbc): A java- based tool to compress dna sequences and compute compression ratio(bits/base) of genomes. CoRR, abs/1006.1193, 2010.
[17] Bharti RK, Verma A, Singh RK. A biological sequence compression based on cross chromosomal similarities using variable length lut. International Journal of Biometrics and Bioinformatics, 4:217–223, 2011.
[18] Mehta A, Patel B. Dna compression using hash based data structure. International Journal of Information Technology and Knowledge Management, July-Dec. 2010, Vo. 2, No. 2, pp. 383-386
[19] Grumbach S, Tahi F. A new challenge for compression algorithms: genetic sequences. Information Processing & Management, 30(6):875–886, 1994.
[20] Chen L, Lu S, Ram J. Compressed pattern matching in dna sequences. In Proceedings of the 2004 IEEE Computational Systems Bioinformatics Conference, CSB'04,
[21] Larsson J, Moffat A. Offline dictionary-based compression. In Proceedings of the 1999 Conference on Data Compression, DCC'99, pages 296–305, 1999.
[22] Shibata Y, Matsumoto T, Takeda M, Shinohara A, Arikawa S. A boyer-moore type algo- rithm for compressed pattern matching. In Proceedings of the 11th Annual Symposium on Combinatorial Pattern Matching, COM'00, pages 181–194, 2000.
[23] Cleary J, Witten I. Data compression using adaptive coding and partial string matching. IEEE Transactions on Communications, 32:396–402, 1984.
[24] Cao MD, Dix TI, Allison L, Mears C. A simple statistical algorithm for biological sequence compression. In Proceedings of the 2007 Conference on Data Compression, DCC'07, pages 43–52, 2007.
[25] Kuruppu S, Puglisi SJ, Zobel J. Relative lempel-ziv compression of genomes for large-scale storage and retrieval. In Proceedings of the 17th International Conference on String Processing and Information Retrieval, SPIRE'10, pages 201–206, 2010.
[26] Kuruppu S, Puglisi S, Zobel J. Optimized relative lempel-ziv compression of genomes. In Australasian Computer Science Conference, 2011.
[27] Ziv J, Lempel A. A universal algorithm for sequential data compression. IEEE Transactions on Information Theory, 23(3):337–343, 1977.
[28] Golomb SW. Run-length encodings. IEEE Transactions on Information Theory, 12:399–401, 1966.
[29] Antoniou D, Theodoridis E, Tsakalidis A. Compressing biological sequences using self adjusting data structures. In Information Technology and Applications in Biomedicine, 2010.
[30] Kaipa KK, Bopardikar AS, Abhilash S, Venkataraman P, Lee K, Ahn T, Narayanan R. Algorithm for dna sequence compression based on prediction of mismatch bases and repeat location. In Bioinformatics and Biomedicine Workshops, BIBMW, 2010.
[31] Hosseini M, Pratas D, Pinho AJ. A Survey on Data Compression Methods for Biological Sequences. doi:10.3390/info7040056, 2016.
[32] Pratas D, Pinho AJ. Compressing the human genome using exclusively markov models. In Miguel P. Rocha, Juan M. Corchado Rodrguez, Florentino Fdez-Riverola, and Alfonso Valencia, editors, PACBB, volume 93 of Advances in Intelligent and Soft Computing, pages 213–220. Springer, 2011.
[33] Venugopal KR, Srinivasa KG, Patnaik L. Probabilistic Approach for DNA Compression, chapter 14, pages 279–289. Springer, 2009.
[34] Tabus I, Korodi G. Genome compression using normalized maximum likelihood models for constrained markov sources. In Information Theory Workshop, 2008.
[35] Chern B, Ochoa I, Manolakos A, Venkat AK, Weissman T. Reference based genome compression. In Information Theory Workshop (ITW), 2012 IEEE, pages 427–431. IEEE, 2012.
[36] Wandelt S, Leser U. Fresco: Referential compression of highly similar sequences. Computational Biology and Bioinformatics, IEEE/ACM Transactions on, 10(5):1275–1288, 2013.
[37] Bonfield JK, Mahoney MV. Compression of fastq and sam format sequencing data. PloS one, 8(3):e59190, 2013.
[38] Ochoa I, Asnani H, Bharadia D, Chowdhury M, Weissman T, Yona G. Qualcomp: a new lossy compressor for quality scores based on rate distortion theory. BMC bioinformatics, 14(1):1, 2013.
[39] Canovas R, Moffat A, Turpin A. Lossy compression of quality scores in genomic data. Bioinformatics, 30(15): 2130–2136, 2014.
[40] Malysa G, Hernaez M, Ochoa I, Rao M, Ganesan K, Weissman T. Qvz: lossy compression of quality values. Bioinformatics, page btv330, 2015.
[41] Ochoa I, Hernaez M, Goldfeder M, Weissman T, Ashley E. Effect of lossy compression of quality scores on variant calling. Briefings in Bioinformatics, doi:10.1093/bib/bbw011, 2016
[42] Numanagić I, Bonfield JK, Hach F, Vges J, Ostermann J, Alberti C, Mattavelli M, Sahinalp SC. The State of the Art in High Throughput Sequencing Data Compression, Nature Methods, doi:10.1038/nmeth.4037

Computational Prediction of Alternative Metabolic Pathways of Plasmodium Falciparum

Jelili Oyelade[1, 2, *], Itunuoluwa Isewon[1, 2], Olufemi Aromolaran[1, 2], Efosa Uwoghiren[1, 2]

[1]Department of Computer & Information Science, Covenant University, Ota
[2]Covenant University Bioinformatics Research Cluster (CUBRe), Ota

Abstract

Plasmodium falciparum (P.f.), the malaria pathogen, has shown substantial resistance to treatment and vaccine thereby requiring urgent, holistic and broad approach to prevent an endemic. Understanding the biology of the malaria parasite has been identified as a vital approach to overcome the threat of malaria. This study reconstructed the *iPfa* Genome-scale metabolic model (GEM) of 3D7 strain of *Plasmodium falciparum* by filling gaps in the model with nineteen (19) metabolites and twenty-three (23) reactions obtained from MetaCyc database. Biomass reactions and exchange reactions were removed since they are mainly used to evaluate changes in flux which is not required in our approach. Twenty (20) currency metabolites were removed from the network because they have been identified to produce shortcuts that are biologically infeasible. The resulting modified *iPfa* GEM was model using reaction graph and a *k*-shortest path algorithm was applied to identify alternative metabolic pathways of *Plasmodium falciparum*. Five alternative paths were predicted in the glycolysis pathway.

Keywords: Graph based technique, *k*-shortest path, *Plasmodium falciparum*, metabolic pathway.

1 Introduction

1.1 The Overview of *Plasmodium falciparum*

Malaria remains one of the leading global health challenge, especially in tropical and subtropical areas, where about 212 million clinical cases and more than 429,000 estimated cases of deaths recorded in the year 2015 [1]. According to World health Organization (WHO), 88% of malaria incidence and death in 2015 are estimated to have occurred in Africa. The major cause of the ailment in human is an organism known as *Plasmodium falciparum (P.f.)*. There exist five variants of the plasmodium clan, these are; *Plasmodium falciparum, Plasmodium vivax, Plasmodium ovale, Plasmodium malariae* and *Plasmodium knowlesi*, out of which *Plasmodium falciparum* and *Plasmodim vivax* are the most dangerous. There have been concerted effort by the governments of Nations and global health body (WHO) in the area of funding, research and drug development to halt the devastating trend of the malaria endemic especially in Africa, however the scourge is still pervasive. Understanding the biology of the malaria parasite has been identified as a vital approach to overcome the threat of malaria [2].

The comprehensive *Plasmodium falciparum* lifecycle comprises of 3 important developmental stages: the mosquito stage, the liver stage, and the blood stage [3]. The malaria parasite metabolic pathways are in a number of ways different from that of a human being because of the uniqueness in the malaria parasite life-cycle, thus it becomes very possible for the malaria parasite to take advantage of the uniqueness of its pathways to design therapeutic strategies [4]–[6] by traversing alternative pathways for its activities which helps the parasite to resisting existing drugs and makes it a core responsibility to discover new drugs[7], [8],[9].

Four major biological networks have been widely studied for the comprehensive understanding of the biology of the malaria parasite, they include: metabolic networks of enzyme catalyzed biochemical reactions between metabolic substrates; protein interaction networks comprising of the physical interactions between an organism's proteins which provides conceptual framework for more insight into the functional organization of the whole complement of proteins that exist within a cell, tissue or organism; transcriptional regulatory networks which depict the regulatory relationships between various genes[10] and the signal transduction network[9]. It enables better understanding how a gene or set of genes determines expression of other genes.

1.2 Metabolic Network

Metabolism can either be catabolism which is the breaking down of complex compound in living organisms to generate smaller ones alongside release of energy which is used by the organism to sustain life or metabolism can be anabolism which is the build-up of complex compounds from simpler ones in living organism. The metabolic network of a specific cell or a living organism is the entire network of metabolic reactions of the cell. In order to make sense of the available metabolic network data, metabolic networks are usually constructed as compound graphs, reaction graphs, enzyme graphs bipartite graph or hyper-graph and a path finding technique is used to enumerate the paths. In this context, we would like to clarify the

difference between a path and a (metabolic) pathway. In graph theory, a path is defined as a linear sequence of nodes connected by edges such that each node pair is connected by only one edge and each path node, including the start and the end node, can be found at most once in a path. However, metabolic pathway is similar to a path but may contain branches, cycles, and multiple instances of the same compound [11].

1.3 Metabolic pathways of *P.f.*

A metabolic pathway is a collection of enzyme catalysed biochemical reactions by which a living organism transforms a source (initial) compound into a target (final) compound. A metabolic pathway could likewise be presented as an interconnected sub-network of the metabolic network either depicting particular activities or characterized by functional limits, e.g., the network between glucose (source substance) and pyruvate (target substance).The theory upon which the path finding approaches is based is that finding directed paths between the source compound and the destination compound in the entire metabolic network will give clear understanding into the intermediate reactions/compounds utilized in the metabolic pathway between the source and destination compounds.Metabolism in eukaryotes is classified into several categories which include; Amino acid metabolism, Nucleic acid metabolism, Carbohydrate metabolism and Lipid metabolism.

1.4 *k*-shortest Path Algorithm

Faust et al., (2010) stated that a simple approach towards the prediction of a pathway is to find the shortest path(s) between a given start and end node in the network. The *k*-shortest paths problem is a network optimization problem where the objective is not only to identify the shortest path, but also the second, the third, the fourth,..., to the k^{th} shortest path from source node *s* to target node *t* in a network. A good number of k-shortest path algorithms such as [12]–[20]list all shortest paths between a given pair of nodes have been applied to several types of networks including biological networks. However, when using them to predict metabolic pathways, two issues specific to metabolic networks need to be considered theses are; reaction directionality and the hub compound problem [11]. Hence, the application of *k*-shortest path technique must considerably consider these two issues and adequately tackle their effects.

2 Materials and Methods

2.1 Reconstruction of the metabolic network

In this study the *iPfa* Genome-scale metabolic model (GEM) [21] which is the latest model was chosen because it represents the most comprehensive, stage-specific *P. falciparum* metabolic reconstruction to date. *iPfa* includes 325 genes and 670 metabolic reactions localized within five intracellular compartments: the nucleus, the apicoplast, the cytosol, the endoplasmic reticulum and the mitochondrion. *iPfa* also accounts for transport reactions: 236 potential uptakes, i.e. transports from the medium (host cell cytosol or blood serum) to the parasite's cytosol, and 155 transports between intracellular compartments. Nineteen (19) metabolites and twenty-three (23) reactions were obtained from MetaCyc database [22] to fill gaps in the *iPfa* GEM while biomass reactions and exchange reactions were removed since they are mainly used to evaluate changes in flux to obtain a modified GEM for the metabolic network. The details about the twenty three (23) reactions obtained from MetaCyc database can be found in Appendix A.

Hub compounds or pool metabolites or currency metabolites are compounds that are commonly involved in metabolic reactions and cause shortcuts without biological meaning when computing paths in a simple graph [23] (e.g. Proton, Water). Kim *et al*., (2015) identified twenty-five (25) currency metabolites out of which we eliminated 20 of such from the network before reconstructing the graph. The remaining five (5) currency metabolites identified by Kim *et al*., (2015) were retained in the network since their presence does not have significant effect that could lead to shortcuts without biological meaning; this therefore partially addresses the issue of hub compounds. Reaction interaction graph was used to model the metabolic network where the vertices are the reactions and the edges represent the interaction between the reactions and uniform weight was assigned to all edges in the network.

2.2 The Algorithm

T-star algorithm was selected among other *k*-shortest paths techniques because of its superior computational performance [22], see table 1. T-star requires a topological sort of the graph nodes as its input. Topological sort is only possible for a Directed Acyclic Graph, hence we implemented the algorithm to avoid loops/cycles. We enhanced the prediction precision of the graph based approach by incorporating a heuristic function which favors paths that are biologically related to the annotated pathway between a given pair of compounds. Our approach is based on the theory that the malaria parasite uses the shortest route to a particular product, hence the assumption that the currently known annotated pathways contain the shortest paths utilized by the pathogen, therefore the heuristic function uses the reactions in the annotated pathways to rank paths that share similar reactions above others. The enhanced T* algorithm was applied to the reconstructed network to obtain *k*-shortest paths for selected metabolic pathways where k = 20.

Table 1: Time Complexity of T*, K*, Yen, Feng, EA and LVEA algorithms.(Source: Kadivar, 2016)

Algorithm	Time Complexity
T*	*O(m + nk log d)*
K*	*O(m + n log n + k)*
LVEA	*O(m+n log n+k log k)*
EA	*O(m+n log n+k log k)*
Feng	*O(kn(m+n log n))*
Yen	$O(1/2(Kn^3))$

The reactions within the annotated pathway for a given biological process such as glycolysis are referred to as the Known nodes in the algorithm shown below.

1: H = [Known nodes]
2: next = source;
3: while next != target do
4: Update(P[next] by P[j], j ∈ A— (next));
5: Reward(P[next] for each h in P[next], h ∈ H)
6: next = next + 1;
7: end while
8: Update(P[i] by P[j], j ∈ A— (i))
9: Make min-priority queue MQ(i) by P1j ∈ P[j] for j ∈ A— (i) according to L(P1j) + c_{ji} values.
10: while MQ(i) ≠ ∅ and |P[i]| < k do
11: Move P at root of MQ(i) to P[i] and replace the moved item by its neighbor in the source list from which it came and update the nodes values traversing up in MQ(i) along the path of the moved item.
12: end while

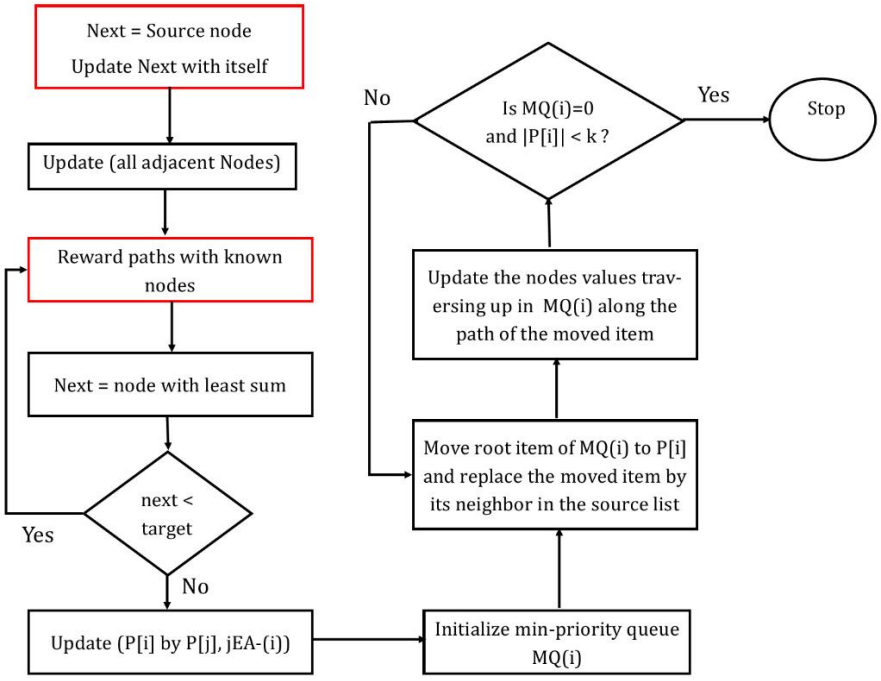


Figure 1: Modified T* Algorithm Flow chart. The boxes labelled in red signifies the point where heuristic function was applied to the algorithm

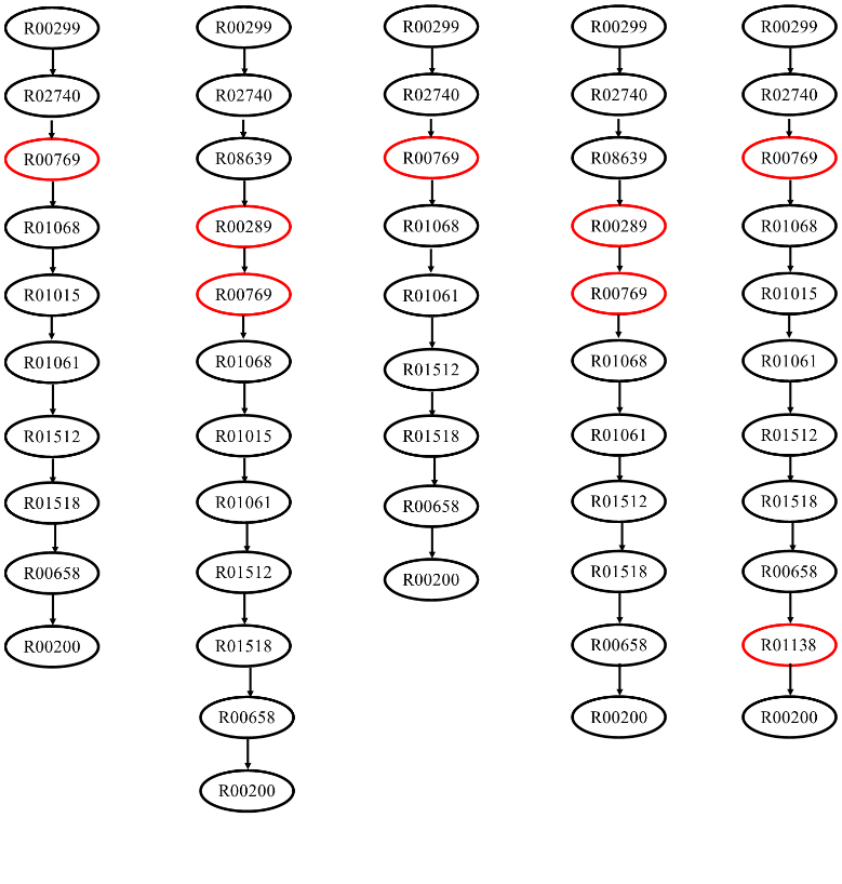
3 Results

We applied our method to the following two metabolic pathways to enumerate *k*-shortest paths, where $k = 5$; Glycolysis Pathway and Pentose Phosphate Pathway.

Eight (8) additional metabolic reactions were predicted to be involved in the glycolysis pathway which could provide an alternative path for conversion of glucose to pyruvate. The list of the 8 reactions are given in Table 2. Also, figure 2 shows the 5 predicted alternative paths for the pathway.

Table 2: Predicted reactions involved in glycolysis pathway

Reaction name in KEGG	Description of Reaction	Associated E.C Number
R00345	phosphate:oxaloacetate carboxy-lyase	4.1.1.31
R00662	Uridine triphosphate pyrophosphohydrolase	3.6.1.8 3.6.1.19
R01138	dATP:pyruvate 2-O-phosphotransferase	2.7.1.40
R00156	ATP:UDP phosphotransferase	2.7.4.6
R08845	UTP:monosaccharide-1-phosphate uridylyltransferase	2.7.7.64
R00769	UTP:D-fructose-6-phosphate 1-phosphotransferase	2.7.1.11
R00341	ATP:oxaloacetate carboxy-lyase (transphosphorylating;phosphoenolpyruvate-forming)	4.1.1.49
R00289	UTP:alpha-D-glucose-1-phosphate uridylyltransferase	2.7.7.9 2.7.7.64



k=1:5

Figure 2: Predicted paths for glycolysis pathway

Five (5) additional metabolic reactions were predicted to be involved in the Pentose Phosphate pathway which could provide an alternative path for conversion of D-Glucose 6-phosphate to D-Glyceraldehyde 3-phosphate. The list of the 5 reactions are given in Table 3. Also, figure 3 shows the 5 predicted alternative paths for the pathway.

Table 3: Predicted reactions involved in Purine metabolism

Reaction name in KEGG	Description of Reaction	Associated E.C Number
R01015	D-glyceraldehyde-3-phosphate aldose-ketose-isomerase	5.3.1.1
R01010	Glycerone phosphate phosphohydrolase	3.1.3.1 3.1.3.2
R00756	ATP:D-fructose-6-phosphate 1-phosphotransferase	2.7.1.11
R02568	D-Fructose 1-phosphate D-glyceraldehyde-3-phosphate-lyase	4.1.2.13
R01068	D-fructose-1,6-bisphosphate D-glyceraldehyde-3-phosphate-lyase (glycerone-phosphate-forming).	4.1.2.13

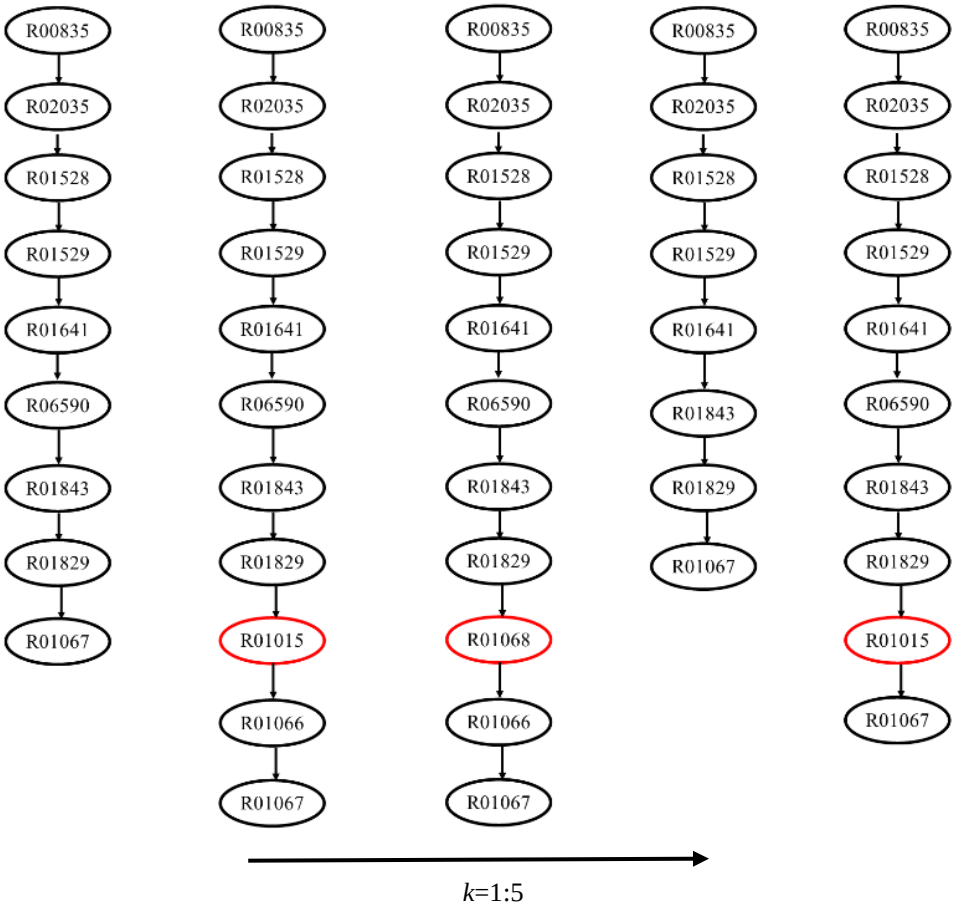


Figure 3: Predicted paths for Pentose Phosphate pathway

4 Discussion

The application of our method on the metabolic network of *P.f.* to predict alternative paths within two (2) metabolic pathways reveals that there are potential reactions that could be involved in the metabolic pathways of the organism that could serve as redundant links between source and destination reactions in the case where the primary link is unavailable or blocked by vaccines. Some of the predicted reactions yet to be validated have red outline in figure 2 and 3. The predicted paths is ordered left to right starting from k=1 to k=5. The implication of these alternative paths is that if path 1 is unavailable for the metabolic activities of the parasite it attempts

to use path 2 and so on and this has been highlighted as the major cause of the drug resistance by the parasite.

5 Conclusion

This study applied T* Algorithm enhanced with heuristic algorithm to predict alternative paths within the metabolic network of *P.f*. The predicted paths could provide valuable insight into the biology of the organism which will aid in effective therapy and vaccine development and subsequently reduce the mortality and morbidity rate of malaria disease. Finally, we observed that dataset used is a GEM prepared for constraint-based modelling and not graph-theoretical path finding approaches which was employed in this study and this has a considerably effect on our prediction.

ACKNOWLEDGMENT

We acknowledge the support of Covenant University Center for Research, Innovation and Discovery, Covenant University, Ota, Nigeria.

6 References

[1] WHO, "World Malaria Report 2016," 2016.

[2] H. Cai, T. G. Lilburn, C. Hong, J. Gu, R. Kuang, and Y. Wang, "Predicting and exploring network components involved in pathogenesis in the malaria parasite via novel subnetwork alignments," *BMC Syst. Biol.*, vol. 9, no. Suppl 4, p. S1, Jun. 2015.

[3] Z. Bozdech, M. Llinás, B. L. Pulliam, E. D. Wong, J. Zhu, and J. L. DeRisi, "The Transcriptome of the Intraerythrocytic Developmental Cycle of Plasmodium falciparum," *PLoS Biol.*, vol. 1, no. 1, p. e5, Aug. 2003.

[4] K. Nakatani, H. Ishikawa, S. Aono, and Y. Mizutani, "Identification of essential histidine residues involved in heme binding and Hemozoin formation in heme detoxification protein from Plasmodium falciparum.," *Sci. Rep.*, vol. 4, p. 6137, Aug. 2014.

[5] E. Hempelmann, "Hemozoin Biocrystallization in Plasmodium falciparum and the antimalarial activity of crystallization inhibitors," *Parasitol. Res.*, vol. 100, no. 4, pp. 671–676, Jan. 2007.

[6] N. Lang-Unnasch and A. D. Murphy, "METABOLIC CHANGES OF THE MALARIA PARASITE DURING THE TRANSITION FROM THE HUMAN TO THE MOSQUITO HOST," *Annu. Rev. Microbiol.*, vol. 52, no. 1, pp. 561–590, Oct. 1998.

[7] G. Plata, T. Hsiao, K. L. Olszewski, M. Llinás, and D. Vitkup, "Reconstruction and flux-balance analysis of the Plasmodium falciparum metabolic network," *Mol. Syst. Biol.*, vol. 6, no. 1, p. 408, 2010.

[8] C. Huthmacher, A. Hoppe, S. Bulik, and H.-G. Holzhütter, "Antimalarial drug targets in Plasmodium falciparum predicted by stage-specific metabolic network analysis," *BMC Syst. Biol.*, vol. 4, no. 1, p. 120, Aug. 2010.

[9] Jelili Oyelade, Ezekiel Adebiyi, Itunu Ewejobi, Benedikt Brors and Roland Eils.(2011) "*Computational Identification of Signalling Pathways in Plasmodium falciparum*". Infection, genetics and evolution, Journal of molecular epidemiology and evolutionary genetics in infectious diseases - Elsevier, Vol. 11(4), 755-764

[10] Itunuoluwa Isewon, Jelili Oyelade,Benedikt Brors, Ezekiel Adebiyi(2015):*In-silico* GeneRegulatory Network of the Maurer's Cleft Pathway in *Plasmodium falciparum.* Evolutionary Bioinformatics. Vol. 11, 231-238

[11] K. Faust, P. Dupont, J. Callut, and J. Van Helden, "Pathway discovery in metabolic networks by subgraph extraction," *Bioinformatics*, vol. 26, no. 9, pp. 1211–1218, 2010.

[12] J. Y. Yen, "Finding the k shortest loopless paths in a network," *Manage. Sci.*, vol. 17, no. 11, pp. 712–716, 1971.

[13] D. Eppstein, "Finding the k shortest paths," *SIAM J. Comput.*, vol. 28, no. 2, pp. 652–673, 1998.

[14] E. Hadjiconstantinou and N. Christofides, "An efficient implementation of an algorithm for finding K shortest simple paths," *Networks*, vol. 34, no. 2, pp. 88–101, 1999.

[15] E. Q. V Martins and M. M. B. Pascoal, "A new implementation of Yen's ranking loopless paths algorithm," *4OR A Q. J. Oper. Res.*, vol. 1, no. 2, pp. 121–133, 2003.

[16] G. Liu and K. G. Ramakrishnan, "A* Prune: an algorithm for finding K shortest paths subject to multiple constraints," in *INFOCOM 2001.*

Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, 2001, vol. 2, pp. 743–749.

[17] A. Y. Hamed, "A genetic algorithm for finding the k shortest paths in a network," *Egypt. Informatics J.*, vol. 11, no. 2, pp. 75–79, 2010.

[18] H. Aljazzar and S. Leue, "K□: A heuristic search algorithm for finding the k shortest paths," *Artif. Intell.*, vol. 175, no. 18, pp. 2129–2154, 2011.

[19] N. Flerova, R. Marinescu, and R. Dechter, "Searching for the M Best Solutions in Graphical Models," *J. Artif. Intell. Res.*, vol. 55, pp. 889–952, 2016.

[20] M. Kadivar, "A new $ O (m+ kn log overline {d}) $ algorithm to Find the $ k $ shortest paths in acyclic digraphs," *Trans. Comb.*, vol. 5, no. 3, pp. 23–31, 2016.

[21] A. Chiappino-Pepe, S. Tymoshenko, M. Ataman, D. Soldati-Favre, and V. Hatzimanikatis, "Bioenergetics-based modeling of Plasmodium falciparum metabolism reveals its essential genes, nutritional requirements, and thermodynamic bottlenecks," *PLOS Comput. Biol.*, vol. 13, no. 3, p. e1005397, 2017.

[22] R. Caspi, T. Altman, R. Billington, K. Dreher, H. Foerster, C. A. Fulcher, T. A. Holland, I. M. Keseler, A. Kothari, and A. Kubo, "The MetaCyc database of metabolic pathways and enzymes and the BioCyc collection of Pathway/Genome Databases," *Nucleic Acids Res.*, vol. 42, no. D1, pp. D459–D471, 2014.

[23] M. Arita, "The metabolic world of Escherichia coli is not small," *Proc. Natl. Acad. Sci. U. S. A.*, vol. 101, no. 6, pp. 1543–1547, 2004.

[24] T. Kim, K. Dreher, R. Nilo-Poyanco, I. Lee, O. Fiehn, B. M. Lange, B. J. Nikolau, L. Sumner, R. Welti, and E. S. Wurtele, "Patterns of metabolite changes identified from large-scale gene perturbations in Arabidopsis using a genome-scale metabolic network," *Plant Physiol.*, vol. 167, no. 4, pp. 1685–1698, 2015.

7 Appendix A

Table 6: Reactions obtained from MetaCyc database to fill gaps in the *iPfa* GEM

NAME	**EQUATION**	**EC-NUMBER**	**SUBSYSTEM**
L-threonine aldolase	L-Threonine[c] <=> Acetaldehyde[c] + Glycine[c]	4.1.2.48, 4.1.2.5	Glycine, serine and threonine metabolism
glycine hydroxymethyltransferase	L-Serine[c] + Tetrahydrofolate[c] <=> Glycine[c] + 5,10-Methylene-tetrahydrofolate[c] + H2O[c]	2.1.2.1	Glycine, serine and threonine metabolism
serine-glyoxylate transaminase	Glyoxylate[c] + L-Serine[c] <=>Hydroxypyruvate[c] + Glycine[c]	2.6.1.45	Glycine, serine and threonine metabolism
ornithine cyclodeaminase	L-Ornithine[c] <=> L-Proline[c] + NH3[c]	4.3.1.12	Arginine and proline metabolism
	(S)-1-Pyrroline-5-carboxylate[c] <=> L-Glutamate 5-semialdehyde[c]		Arginine and proline metabolism
aspartate 4-decarboxylase	L-Aspartate[c] <=> L-Alanine[c] + CO2[c]	4.1.1.12	Alanine, aspartate and glutamate metabolism
citrate (Si)-synthase	Acetyl-CoA[c] + H2O[c] + Oxaloacetate[c] <=> Citrate[c] + CoA[c]	2.3.3.1	Alanine, aspartate and glutamate metabolism
glutamine-pyruvate transaminase	L-Glutamine[c] + Pyruvate[c] <=> 2-Oxoglutaramate[c] + L-Alanine[a]	2.6.1.15	Alanine, aspartate and glutamate metabolism
2-oxoglutaramate amidase	2-Oxoglutaramate[c] + H2O[c] <=> 2-Oxoglutarate[c] + NH3[c]	3.5.1.111	Alanine, aspartate and glutamate metabolism
cystathionine gamma-synthase	L-Cysteine[c] + O-succinyl-L-homoserine[c] <=> succinate[c] + L-cystathionine[c] + H+[c]	2.5.1.48	Cysteine and methionine metabolism
cystathionine beta-lyase	L-cystathionine[c] + H2O[c] <=> ammonium[c] + pyruvate[c] + L-homocysteine[c]	4.4.1.8	Cysteine and methionine metabolism
methionine synthase	L-homocysteine[c] + N5-methyltetrahydrofolate[c] <=> L-Methionine[a] + tetrahydrofolate[c]	2.1.1.13	Cysteine and methionine metabolism
5-methyltetrahydropteroyltriglutamate-homocysteine S-methyltransferase	L-homocysteine[c] + N5-methyltetrahydropteroyl tri-L-glutamate[c] <=> L-Methionine[a] + tetrahydropteroyl tri-L-glutamate[c]	2.1.1.14	Cysteine and methionine metabolism
cystathionine gamma-lyase	L-cystathionine[c] + H2O[c] <=> 2-oxobutanoate[c] + L-Cysteine[c] + ammonium[c]	4.4.1.1	Cysteine and methionine metabolism

inositol-tetrakisphosphate 5-kinase	D-myo-inositol (1,3,4,6)-tetrakisphosphate[c] + ATP[c] <=> D-myo-inositol 1,3,4,5,6-pentakisphosphate[c] + ADP[c] + H+[c]	2.7.1.140	Inositol phosphate (vit B8) metabolism
inositol-1,3,4-trisphosphate 5/6-kinase	D-myo-inositol (1,3,4)-trisphosphate[c] + ATP[c] <=> D-myo-inositol (1,3,4,6)-tetrakisphosphate[c] + ADP[c] + H+[c]	2.7.1.159	Inositol phosphate (vit B8) metabolism
inositol-1,3,4-trisphosphate 5/6-kinase	D-myo-inositol (1,3,4)-trisphosphate[c] + ATP[c] <=> D-myo-inositol (1,3,4,5)-tetrakisphosphate[c] + ADP[c] + H+[c]	2.7.1.159	Inositol phosphate (vit B8) metabolism
inositol-1,4-bisphosphate 1-phosphatase	D-myo-inositol (1,3,4)-trisphosphate[c] + H2O[c] <=> D-myo-inositol (3,4)-bisphosphate[c] + phosphate[c]	3.1.3.57	Inositol phosphate (vit B8) metabolism
phosphatidylinositol-3,4-bisphosphate 4-phosphatase	D-myo-inositol (3,4)-bisphosphate[c] + H2O[c] <=> 1D-myo-Inositol 3-phosphate[c] + phosphate[c]	3.1.3.66	Inositol phosphate (vit B8) metabolism
inositol-polyphosphate multikinase	D-myo-inositol (1,4,5)-trisphosphate[c] + ATP[c] <=> D-myo-inositol (1,4,5,6)-tetrakisphosphate[c] + ADP[c] + H+[c]	2.7.1.151	Inositol phosphate (vit B8) metabolism
inositol-polyphosphate multikinase	D-myo-inositol (1,4,5,6)-tetrakisphosphate[c] + ATP[c] <=> D-myo-inositol 1,3,4,5,6-pentakisphosphate[c] + ADP[c] + H+[c]	2.7.1.151	Inositol phosphate (vit B8) metabolism
inositol-trisphosphate 3-kinase	D-myo-inositol (1,4,5)-trisphosphate[c] + ATP[c] <=> D-myo-inositol (1,3,4,5)-tetrakisphosphate[c] + ADP[c] + H+[c]	2.7.1.127	Inositol phosphate (vit B8) metabolism
inositol-pentakisphosphate 2-kinase	D-myo-inositol 1,3,4,5,6-pentakisphosphate[c] + ATP[c] <=>phytate[c] + ADP[c] + H+[c]	2.7.1.158	Inositol phosphate (vit B8) metabolism

Identifying Temporal Variation of Transcription in Populations

Aisharjya Sarkar, Prabhat Mishra and Tamer Kahveci
Department of Computer and Information Science and Engineering, University of Florida
Gainesville, Florida, 32611, USA
(aisharjya, prabhat, tamer)@ufl.edu

Abstract

We consider the problem of aligning multiple time series gene expression data with the goal of achieving minimum SP (sum of pairwise) distance score. The transcription level of genes often change over time. For a gene, the rate of change of transcription values can vary across different members of a population. Therefore, alignment of such data has great potential to reveal how the cellular activities of different samples deviate from each other. We develop an algorithm for alignment of multiple time series expression data using dynamic time warping approach. Our experiments demonstrate that our method always improves the alignment score. We observe that the correlation between significant gene pairs show drastic improvement after our method is applied, which helps in uncovering key functional characteristics from biological perspective. Furthermore, the alignment strategy is independent of the length of different time series under consideration.

keywords: gene expression, time series, multiple alignment, dynamic time warping

1 Introduction

Genes are segments within DNA which contains within themselves instructions for producing molecules to carry out various functions within cells. The process by which the information contained within a gene transforms into the product called RNA is called transcription or expression. Typically not all gene products are needed all the time. Even when a specific set of gene products are needed, the amount of time it takes to produce them and the rate at which a gene produces them varies. In fact, it will be energy inefficient for a cell to express every gene all the time. Further, some gene products are harmful to the cell if it is not produced at the right time or quantity. Recent technology such as high throughput sequencing makes it possible to record expression values for thousands of genes in parallel by means of gene expression profiling. This results in huge amount of transcription information.

Gene expression data is usually represented in a matrix or tabular form where each row represents a gene or probe. Each column typically represents a sample. Each entry of this matrix is the expression level of a particular gene for a particular sample. Several existing efforts has already been done to analyze such data in order to extract meaningful biological knowledge [3]. Most of the studies on transcriptome are limited to static expression analysis, where a value is a snapshot of the expression of genes in different samples at a particular time point. There are also studies which measure the transcription values of a sample multiple times over a period of time [17]. We call such datasets as time series gene expression data. Studying of time series data is extremely important since it has great potential to reveal the process through which cells react to transcriptional variations, external stimulants (such as drugs), or major disorders (such as cancer) over a period of time.

One major hurdle in studying time series expression data arises from inherent characteristics of cells. The genes of different samples (eg., different individuals) often vary in their rates at which they react to transcriptional alterations. For instance, the same gene from a young and old individual may have different reaction rate. Also different samples may possess varying genetic or epigenetic mutations or may be subject to external stimulants (such as drugs) or they may be at different stages of a disease. As a result of such variation, the specific time points in time series expression of a sample may not match to the same time point in the time series expression of another sample. Thus, finding which time point in a time series sample correspond to which time point(s) in a given set of time series sample is essential in studying such data. As expression levels can be measured across time in different organisms including humans, it is useful to compare the expression changes across different time points between two or more time series data. Since the common biological processes may function at varying rates in different individuals or organisms, methods are required that will allow to map the expression states between different time series. Such analyses reveal

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

temporal features of gene expression changes, such as acceleration or delay in transcription values. This temporal change in gene expression values play a vital role in the biology of an organism since the information from a gene is used in the synthesis of proteins. One such example is inter-species time series expression data comparison [5]. In this work, the authors compared the similarities between temporal expression data between two species (*Saccharomyces cerevisiae*, *Schizosaccharomyces pombe*) that are separated by more than 400 million years of evolution by means of global alignment.

There are several methods that employed the concept of dynamic time warping (DTW) to address temporal alignment problems. DTW is a type of time series alignment algorithm for measuring similarity between two time series data that may vary in speed. DTW was originally developed to address speech recognition problems [13, 14]. Aach and Church [2] introduced the concept of DTW for gene expression data. This has been further developed by other groups [15, 10, 16]. Although there is a substantial amount of research on DTW, to the best of our knowledge they are limited to comparing only two time series [12]. 'Continuous Profile Model' [9] was primarily developed for speech waveform time series data where frequent sampling of data is a reasonable approach. Here, smoothness in time is necessary for the 'continuous' CPM alignment which is not feasible for real time gene expression data. Therefore, it is not appropriate to apply CPM alignment to the breast cancer dataset considered here.

In this paper, we develop a method to tackle the problem of alignment of multiple time series expression data using the DTW approach. Our method adopts the progressive alignment strategies used for aligning multiple nucleotide or amino acid sequences [18, 11, 7] to DTW of multiple transcription data. We define a score-based progressive alignment algorithm by means of dynamic programming technique on successive branches of a guide tree. Guide tree decides the order of alignment in the progressive multiple alignment heuristic. It finds the optimal alignment between a pair of one or more than one time series with respect to a given scoring function. This study addresses two major challenges that are inherent characteristics of time series data:

- Often time series data are sampled at non-uniform time intervals. The time points observed in one time series may not correspond to measured time points in another time series.
- The time series may be of variable size. One time series data may contain fewer observations than another.

We organize the rest of the paper as follows. We present our algorithm in Section 2. We discuss our experimental results in Section 3 and conclude in Section 4.

2 Methodology

We develop an algorithm to align multiple gene expression time-series data. By alignment of gene-expression data, we mean a mapping from time coordinates in each series to those of the remaining ones such that the expression values at those time points are similar. In this section, we present a detailed description of our method. We present the basic notation we use in this paper in Section 2.1. We then describe two distance measures. Section 2.2 discusses how we measure the distance while allowing flexibility to the amount of time in order to capture the best possible mapping or alignment between a pair of time-series data. We then elaborate on each step of our method, which works for multiple time-series data, in detail in Section 2.3.

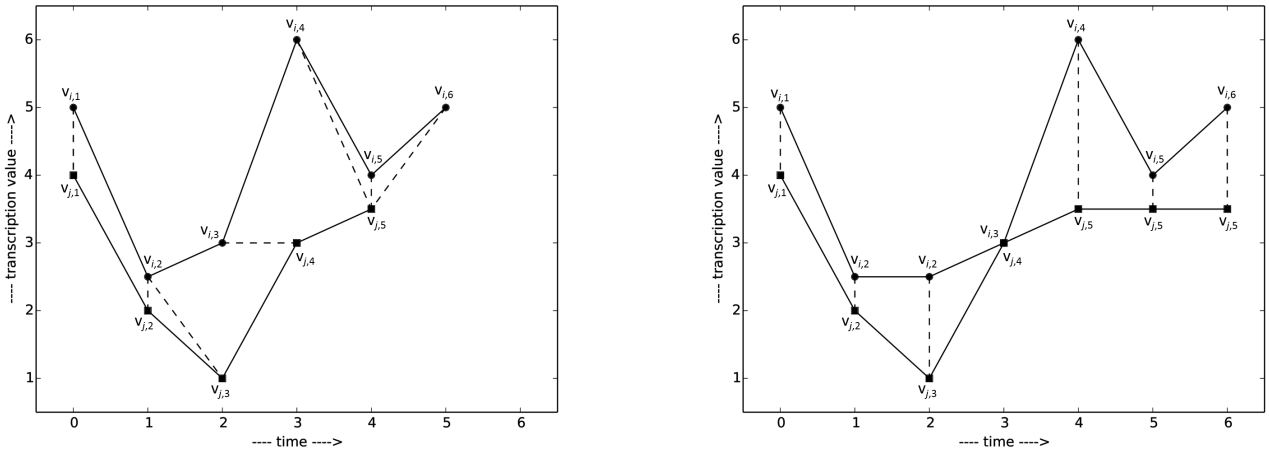
2.1 Notation

We start by describing the notation that will be used in the rest of this paper. Let us assume that a given temporal dataset contains m samples, where each sample is the expression level of a given gene of a patient measured over a period of time. We denote the samples in the given dataset with $S_1, S_2, \cdots, S_m$. Let us denote the number of time points at which transcription of the given gene is measured for the ith sample with n_i. We represent each sample S_i using a vector as $S_i = [v_{i,1}, v_{i,2}, \cdots, v_{i,n_i}]$, where each entry $v_{i,j}$ shows the transcription level of the gene under consideration belonging to the ith sample observed at the jth time point.

2.2 Measuring distance between time series data

Consider two time series samples S_i and S_j. Also, consider the following two assumptions: (i) S_i and S_j have the same number of measurements (i.e., $n_i = n_j$), and (ii) for all time points k, where $1 \leq k \leq n_i$, the measurement taken at the kth time point of S_i ($v_{i,k}$) corresponds to that of S_j ($v_{j,k}$). Under these two assumptions, the distance between S_i and S_j is often computed as a simple vector norm of the difference $S_i - S_j$. First and second vector norms are among the most commonly used distance measure in the literature. These norms are also known as the *Manhattan* and the *Euclidean* distances, and are computed as $\sum_k |v_{i,k} - v_{j,k}|$ and $\sqrt{\sum_k (v_{i,k} - v_{j,k})^2}$, respectively.

Although the vector norms discussed above are used



(a) Aligned dynamic states

(b) Stretched dynamic states

Figure 1: An illustration of time-warping of two hypothetical samples, S_i and S_j. Each circle (square) represents a time point. The time points of S_i are labeled with values from $v_{i,1}$ to $v_{i,6}$. The time points of S_j is labeled with values from $v_{j,1}$ to $v_{j,5}$. In left figure 1a the two solid line curves show the actual patterns of the two dynamic states. Dashed lines show the time-warping alignment of the two dynamic states. The right figure 1b shows the time points that are aligned with multiple time points from the other states are duplicated and stretched to match them along the time axis.

commonly for different time series datasets, they cannot easily be used for gene expression time series data. This is because the two underlying assumptions leading to these norms do not hold when applied to gene expression sequences for several reasons. First, the transcription measurements of different samples can be taken at different number of time points (i.e., it is possible to have $n_i \neq n_j$). Second and more importantly even when $n_i = n_j$, two samples can exhibit longitudinal variance. In other words, the kth time point of S_i does not necessarily represent the same state of S_j at the kth time point as the same gene in two different samples can have different response time to transcriptional regulation. Such variation in response time happens for many reasons. Gene copy number variations and epigenetic mutations are only two examples to those reasons. Regardless of the biological reason, the net outcome observed is that in such scenarios, the transcriptional values of two samples can have similar values but the time it takes to reach this value may differ. We say that such times series expression values are *stretched* along the time axis.

Time-warping distance takes into account temporal distortions. Briefly, time-warping distance aligns two given temporal vectors in order to find a mapping between their time points. It does this by stretching them to bring their similar values close to each other. Figure 1 shows the alignment of two hypothetical time series data using time-warping distance. In this example, the time series on the top contains values for six time points labeled from $v_{i,1}$ to $v_{i,6}$. The other time series has values for five time points labeled from $v_{j,1}$ to $v_{j,5}$. In Figure 1a, the dashed lines between two time points, one from each time series, show the aligned time points. Notice that one time point of one series can align with multiple time points of the other. We say that the measurements at such time points are stretched along the time axis. We represent this process by making multiple copies of such values consecutively. Figure 1b shows how the two time series data are stretched along the time dimension to match their time points. For instance, in Figure 1a, $v_{i,2}$ is aligned with both $v_{j,2}$ and $v_{j,3}$. We reflect this in Figure 1b by making two copies of $v_{i,2}$. Notice that, after stretching the two sequences, they both have the same number of time points. Once the two dynamic states are aligned, this measure computes the distance as the Manhattan distance between the two stretched time series data. As we stretch the time series, the number of time points grow. To accomodate for this change, we normalize the distance between two time series data by dividing their Manhattan distance with the number of time points in the stretched time axis.

There are two major challenges in computing time warping distance is to determine which time points will be stretched and by how much. We address these challenges by using dynamic programming technique to align two given time series data as follows. Let us denote the distance between the first k values of S_i and the first r values of S_j with $\gamma(S_i, k, S_j, r)$. We compute this function as follows:

$$\gamma(S_i, k, S_j, r) = |v_{i,k} - v_{j,r}| + min\{\gamma(S_i, k-1, S_j, r-1), \gamma(S_i, k-1, S_j, r), \gamma(S_i, k, S_j, r-1)\} \quad (1)$$

The *min* function consists of the following three scenarios:

- The first scenario arises when the first $k-1$ values of S_i are aligned to the first $r-1$ values of S_j.
- The second scenario corresponds to the case that the first $k-1$ values of S_i are aligned to the first r values of S_j. In other words, S_j is *stretched* along the time axis.
- The third scenario occurs when the first k values of S_i are aligned to the first $r-1$ values of S_j. That is, S_j is *contracted* along the time axis.

Let $k \in [1:k]$, which denotes the first k values of S_i and $r = 1$, which denotes the first value of S_j. Then there is only one possible warping path between $S_i(1:k)$ and $S_j(1:1)$ having a total distance as

$$\gamma(S_i, k, S_j, 1) = \sum_{n=1}^{k} |v_{i,n} - v_{j,1}|$$

This determines the stopping criteria in order to minimize an expected cost. Similarly, we compute the optimal distance between S_i when $k = 1$ and S_j when $r \in [1:r]$, which represents the first r values of S_j as

$$\gamma(S_i, 1, S_j, r) = \sum_{n=1}^{r} |v_{i,1} - v_{j,n}|$$

2.3 Algorithm

In this section, we present our method in detail. Our method takes a set of m vectors, where each vector S_i represents a sample having transcription values for n_i time points as input. It works in three steps as follows: (i) Generation of initial distance matrix $\mathcal{D}$. (ii) Construction of guide tree $\mathcal{T}$. (iii) Construction of alignment matrix $\mathcal{A}$

2.3.1 Generating initial distance matrix

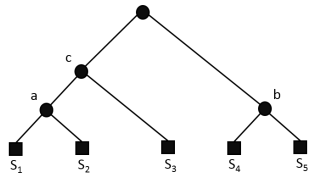
Given a set of m vectors, we construct an $m \times m$ matrix called the distance matrix and denote it with $\mathcal{D}$. This is an upper triangular matrix. For all $1 \leq i < j \leq m$, $\mathcal{D}[i,j]$ is equal to the time warping distance between S_i and S_j. We do not need to store the diagonal of the matrix as the distance between a time series with itself is always zero. Also we do not store its lower triangular part as the time warping distance is commutative (i.e., $\mathcal{D}[i,j] = \mathcal{D}[j,i], \forall i \neq j$). Figure 2a shows a hypothetical distance matrix $\mathcal{D}_5$ consisting of five time series S_1, S_2, S_3, S_4 and S_5. Each cell within the matrix consists of a pairwise time warping distance between two time series. For example, the time warping distance between two time series S_1 and S_3 is represented as d_{13}.

2.3.2 Constructing guide tree

Once we generate the initial distance matrix $\mathcal{D}$, we construct a tree called the *guide tree* using $\mathcal{D}$. Guide tree is a rooted and bifurcating tree (i.e., each internal node has two child nodes). It determines the ordering at which we align the time series samples. More specifically, *guide tree* aims to order the alignment of the entire dataset of m time series in a way such that the closely related series are aligned before the relatively distant ones. Figure 2b illustrates a hypothetical *guide tree*, where each leaf node represents a unique time series sample. Therefore, S_1, S_2, S_3, S_4 and S_5 are the time series represented by the leaf nodes in Figure 2b. Next we discuss in detail how we generate this *guide tree*.

	S_1	S_2	S_3	S_4	S_5
S_1		(1.07)	2.6	3.2	3.5
S_2			2.4	2.9	3.1
S_3				2.7	2.9
S_4					1.6
S_5					

(a) Initial distance matrix consisting of five time series S_1, S_2, S_3, S_4 and S_5.



(b) Guide Tree

Figure 2: A hypothetical initial distance matrix and guide tree.

We build *guide tree* in an iterative manner with the help of the distance matrix $\mathcal{D}$. Each iteration contains two steps:

(1) We pick the pair of samples (S_i, S_j) with the least pairwise distance. This corresponds to the entry $\mathcal{D}[i,j]$ in $\mathcal{D}$ with the smallest value among all $i < j$. If there are multiple such sample pairs with the same value, we pick one arbitrarily. We then construct an internal node of the guide tree which takes S_i and S_j as two children.

(2) We remove S_i and S_j from the sample set and replace them with the new hypothetical sample corresponding to their alignment. We explain how we align them in Section 2.3.3. This reduces the number of samples by one. We update the distance matrix accordingly by computing the distance between the new sample and all the remaining samples.

At the end of each iteration we construct one internal node of the *guide tree*. More specifically, the new internal node is connected to the two nodes corresponding to the two time series S_i and S_j selected at that step. The new internal node denotes the new hypothetical time series generated at that iteration from S_i and S_j. We repeat this process until we have only one time series left. The final (hypothetical) time series denotes the root of the *guide tree*.

2.3.3 Building alignment matrix

Once we complete the construction of the guide tree, we are ready to align all the time series expression data.

We do this iteratively. Let us denote the given set of m time series data with $\mathcal{S} = \{S_1, S_2, \cdots, S_m\}$. Recall that each leaf level node of the guide tree contains a time series data from the set $\mathcal{S}$. Briefly, at each iteration, we pick an internal node of the guide tree such that both children of that node contains a time series expression data or an alignment of a set of time series data. We then align the two time series data corresponding to it's children using dynamic time warping and place the resulting alignment at that internal node. Thus at the end of each iteration, we populate one internal node of the guide tree with the alignment of all the time series data located at the leaf nodes of the clade rooted at that internal node. We repeat this process until we reach the root node of the guide tree. The alignment contained at the root node is the multiple alignment of the entire time series dataset.

Notice that, the internal node selected at each iteration falls into one of the three possible scenarios depending on it's two children.

- Both children nodes contain single time series data. This happens when both children are leaf nodes.
- One child node contains a time series data and the other contains an alignment of a set of time series. This happens when one child is a leaf node and the other is an internal node.
- Both children nodes contain the alignment of a set of time series data. This happens when both children are internal nodes.

Of the above three scenarios, the first one is straightforward. We use the dynamic programming method described in Section 2.2 to align the given pair of time series data. The challenge lies in the remaining two cases, where, at least one of the two children is the alignment of a set of time series. Notice that the second case is special form of the third one since a time series can be considered as a set containing only one series. In the following, we explain how we address the third case.

Our solution follows from the observation that each internal node represent a set of time series that are already aligned. It preserves this alignment while aligning that set with another. Consider an internal node T_i of the guide tree $\mathcal{T}$. Let us denote the set of time series contained in the clade rooted at T_i with

$$\mathcal{S}_i = \{S_{\pi_1}, S_{\pi_2}, \cdots, S_{\pi_i}\}, \forall \mathcal{S}_i \subseteq \mathcal{S}$$

and $\pi_1, \pi_2, \cdots, \pi_i \in \{1, 2, \cdots, m\}$. For instance, for the guide tree in Figure 2b, the set of time series for the internal node c is $\mathcal{S}_c = \{S_1, S_2, S_3\}$. Recall that the time warping distance potentially stretches the time series in set $\mathcal{S}_i$ while aligning them. We denote the alignment of the time series in $\mathcal{S}_i$ with matrix $\mathcal{A}_i$ as

$$\mathcal{A}_i = \begin{bmatrix} v_{\pi_1,\sigma_{1,1}} & v_{\pi_1,\sigma_{1,2}} & \cdots & v_{\pi_1,\sigma_{1,p}} \\ v_{\pi_2,\sigma_{2,1}} & v_{\pi_2,\sigma_{2,2}} & \cdots, & v_{\pi_2,\sigma_{2,p}} \\ \vdots & \vdots & \ddots & \vdots \\ v_{\pi_i,\sigma_{i,1}} & v_{\pi_i,\sigma_{i,2}} & \cdots & v_{\pi_i,\sigma_{i,p}} \end{bmatrix}$$

Each row in this notation denotes a time series in $\mathcal{S}_i$ after it is aligned with the rest of the time series in that set. For instance, the first row denotes S_{π_1}, the second one denotes S_{π_2}, and so on. The subscript $\sigma_{i,k}$ is a monotonically non-decreasing value with k ($\forall k, \sigma_{i,k} \leq \sigma_{i,k} + 1$). It refers to the index of a particular value within the time series S_{π_i} after stretching it. For simplicity, we will refer to each entry $v_{\pi_i,\sigma_{i,k}}$ in the matrix $\mathcal{A}_i$ as $v_{r,s}$, where r denotes the time series index and s denotes the position index within a time series after stretching it. We denote the entry in the r^{th} row and s^{th} column of $\mathcal{A}_i$ as $\mathcal{A}_i[r][s] = v_{r,s}$. For instance, in Figure 2b, we express the alignment of the time series in $\mathcal{S}_c = \{S_1, S_2, S_3\}$, rooted at the internal node c with the matrix $\mathcal{A}_c$ as

$$\mathcal{A}_c = \begin{bmatrix} v_{1,1} & v_{1,2} & \cdots & v_{1,p} \\ v_{2,1} & v_{2,2} & \cdots & v_{2,p} \\ v_{3,1} & v_{3,2} & \cdots & v_{3,p} \end{bmatrix}$$

Notice that $\mathcal{A}_i$ is the partial alignment of all the time series in set $\mathcal{S}_i$.

Let us consider another subset of j time series $\mathcal{S}_j$, where $\mathcal{S}_j \subseteq \mathcal{S}_m \setminus \mathcal{S}_i$. Let $\mathcal{A}_j$ be the alignment matrix of the times series in $\mathcal{S}_j$. Next, we explain how we align the two sets $\mathcal{S}_i$ and $\mathcal{S}_j$.

When we align two internal nodes i and j in the guide tree, it means we are trying to align a partial alignment $\mathcal{A}_i$ with that of $\mathcal{A}_j$. The following equation calculates the distance between first q values of $\mathcal{A}_i$ with that of first r values of $\mathcal{A}_j$ by means of dynamic programming technique as

$$\gamma(\mathcal{A}_i, q, \mathcal{A}_j, r) = \sum_{x=1}^{|\mathcal{S}_i|} \sum_{y=1}^{|\mathcal{S}_j|} |v_{\pi_x,q} - v_{\pi_y,r}| + min\{\gamma(\mathcal{A}_i, q-1, \mathcal{A}_j, r-1), \gamma(\mathcal{A}_i, q-1, \mathcal{A}_j, r), \gamma(\mathcal{A}_i, q, \mathcal{A}_j, r-1)\} \quad (2)$$

In matrix notation, we rewrite the above equation as

$$\gamma(\mathcal{A}_i, q, \mathcal{A}_j, r) = \sum_{x=1}^{|\mathcal{S}_i|} \sum_{y=1}^{|\mathcal{S}_j|} |\mathcal{A}_i[x][q] - \mathcal{A}_j[y][r]| + min\{\gamma(\mathcal{A}_i, q-1, \mathcal{A}_j, r-1), \gamma(\mathcal{A}_i, q-1, \mathcal{A}_j, r), \gamma(\mathcal{A}_i, q, \mathcal{A}_j, r-1)\} \quad (3)$$

Notice that this equation follows from Equation 1. The main difference is that the first term considers all combination of sequence pairs instead of one pair. The following equation shows the stopping condition, which

aligns q values for each time series of $\mathcal{A}_i$ with one value for each time series of $\mathcal{A}_j$.

$$\gamma(\mathcal{A}_i, q, \mathcal{A}_j, 1) = \sum_{n=1}^{q} (\sum_{x=1}^{|\mathcal{S}_i|} \sum_{y=1}^{|\mathcal{S}_j|} |\mathcal{A}_i[x][n] - \mathcal{A}_j[y][1]|) \quad (4)$$

Computation of final alignment score. Let $\mathcal{A}_m$ be the final alignment matrix for the given set of m time series data, $\mathcal{S} = \{S_1, S_2, \cdots, S_m\}$. As discussed above, each row in $\mathcal{A}_m$ represents a time series stretched to align with the other time series in the dataset $\mathcal{S}$, therefore, a total of m rows. Let, the stretched version of $\mathcal{S}$ has a total of p columns. $\mathcal{A}_m$ is shown below. Each element $v_{i,j}$ represents the transcription value of the i_{th} row and j_{th} column.

$$\mathcal{A}_m = \begin{bmatrix} v_{1,1} & v_{1,2} & \cdots & v_{1,p} \\ v_{2,1} & v_{2,2} & \cdots, & v_{2,p} \\ \vdots & \vdots & \ddots & \vdots \\ v_{m,1} & v_{m,2} & \cdots & v_{m,p} \end{bmatrix}$$

The final alignment score of $\mathcal{A}_m$ is calculated as the total of the sum of the pairwise distance between all possible pairs per column, for all columns, divided by the number of columns. The formulation of alignment score is as below:

$$SPdist = 1/p(\sum_{z=1}^{p} \sum_{\substack{x,y=1 \\ x<y}}^{m} |v[x][z] - v[y][z]|) \quad (5)$$

Here after, we refer to the final alignment score as SP (sum of pairwise) distance.

3 Experimental Results

In this section, we extensively evaluate the performance of our method on a real breast cancer dataset. We measure the performance in terms of both quantitative and qualitative analysis. In the following, we describe in detail about the dataset and the analysis on experiments. We have tested our method on simulated datasets as well described in [1].

3.1 Experimental Setup

We use the breast cancer dataset (DataSet Record: GDS4088) from Gene Expression Omnibus (GEO) dataset [6]. This dataset provides transcription values of breast tumor samples preserved using two different RNA stabilization methods. Time series of eleven breast cancer samples subjected to different cold ischemic stress of up to 3 hour post tumor excision is reported in this dataset. The dataset contains tumor tissue samples that are collected from 11 previously untreated breast cancer patients at surgery and divided into 8 portions. For our analysis, we consider the dataset on RNAlater stabilization method which provides values for 6 time points (baseline, 20, 40, 60, 120 and 180 minutes) accross 11 patients. This dataset contains the time series gene expression values for 21,775 genes. Many genes have multiple time series data arising from multiple probes in the dataset. After filtering of the duplicate genes we have 13,630 unique genes in the dataset. In order to select genes relevant to breast cancer, we referred to 12 datasets on breast cancer for *Homo sapiens* from Molecular Signatures Database (MSigDB) [8]. We took union of all the genes from these 12 datasets to form a larger dataset of 1,842 genes. We intersect this dataset with the 13,630 unique genes in our breast cancer dataset leading to 1,412 genes. This filtering enables us to select the most relevant genes in this scenario. We organize each gene in a dataset as a matrix where each row represents a sample of a particular patient and each column denotes a time point in increasing order. Each value in this matrix is the gene expression value at a certain time point for that sample.

3.2 Quantitative analysis

In order to quantitatively analyze our method, we first pre-process the data where we select a subset of relevant genes from the entire set of more than 13,500 genes originally produced in this dataset. We employ two mechanisms: *statistical* pre-processing and *domain-specific* (biological) pre-processing for this purpose.
Statistical pre-processing filters the genes whose expressions do not change significantly over time. This is because, alignment of the transcription of such genes is trivial as all values are almost identical. Furthermore, we conjecture that such genes are less likely to be associated with the disease progression as their values remain unchanged over time. To do this, we calculate the coefficient of variation of the gene expression values of each gene per patient across 6 given time points. The coefficient of variation cv for a set of observations is defined as standard deviation σ of the observations divided by their mean μ. We calculate coefficient of variation for each gene per patient as $cv = \sigma/\mu$. The coefficient of variation gives a measure of the spread of expression values that describes the amount of variability relative to the mean. We get a total 11 such values of cv per gene, one for each of 11 patients. Next, we calculate the average of cv of each gene for all the 11 patients to get a single value for each gene. We perform this operation for all the genes in the dataset and rank the entire gene list in descending order of this measure. We set a cut-off value at 0.25. We select all the genes having coefficient of variation above this cut-off. This results in 681 genes.

For *biological pre-processing*, we collect 12 gene sets

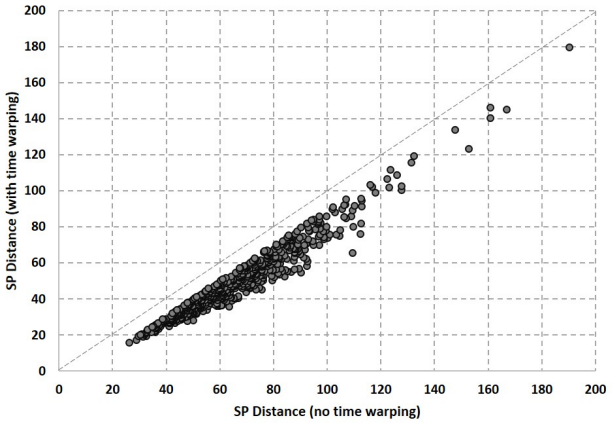


Figure 3: Comparison of sum of pairwise distance (SP distance) between no time warping (along x-axis) and our method (with time warping, along y-axis) on the 1,412 biologically significant genes. Our method always yields a lower SP distance compared to no time warping.

related to breast cancer from **CGP** (chemical and genetic perturbations), **C2**(curated gene sets) of MSigDB (Molecular Signatures Database) [8]. There are in total 1,842 unique genes in this dataset. An intersection of this set of genes extracted from MsigDB with that of the entire time-series dataset of more than 13,500 genes resulted in a filtered dataset of 1,412 genes. Whereas, an intersection of the same set of 1,842 breast cancer related genes with that of the 681 genes extracted by means of statistical pre-processing (explained before) resulted in 39 genes.

In this manner, we build an initial subset of 2,054 relevant genes (681 statistical significant genes + 1,412 biologically significant genes - 39 common genes). Figure 3 shows a scatter plot where each dot represents a gene from the biologically significant geneset of 1,412 genes. We plot along x-axis and y-axis, the SP distance distance without and with time-warping, respectively. It is seen from the figure that our method with time-warping always yield a lower SP distance. Among the biologically significant genes, we observe a significant reduction of 40% to 20% on SP distance after time-warping. Figure 4 shows SP distance of the top 20 genes with highest SP gain among all statistically significant genes. Per gene, it shows SP distance with and without time warping. Here too our method dramatically reduces the SP distance cost over the alignment without time warping. Our algorithm reduces the SP distance by a maximum of 46%.

3.3 Qualitative results

Pairwise gene correlation: Correlation can be viewed as the relationship between transcription levels of gene pairs across different samples. Thus high correlation is observed when time points of different samples are aligned correctly. We perform this experiment in order

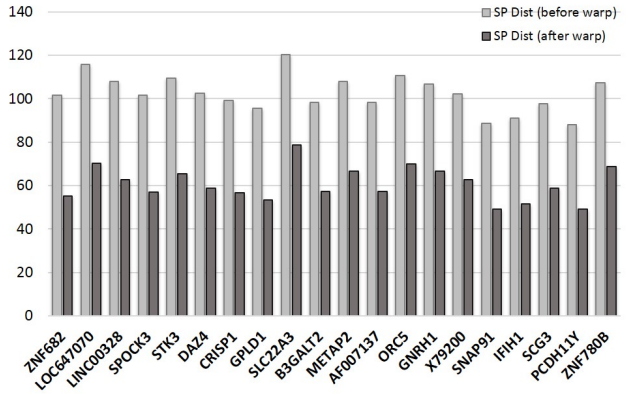


Figure 4: Sum of pairwise distance (Genes retrieved from statistical pre-processing).

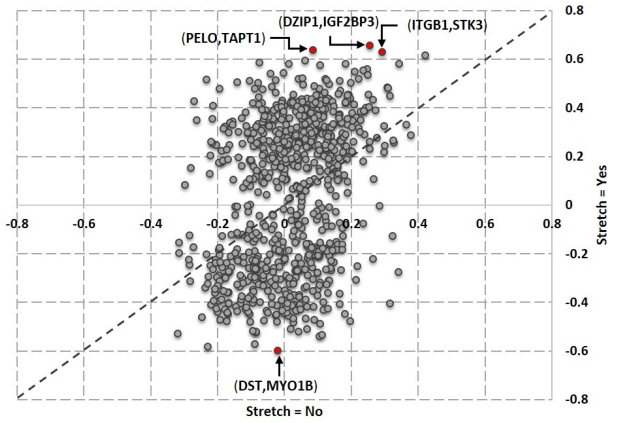


Figure 5: Pairwise gene correlation for 39 genes (statistical intersect biological geneset).

to test the improvement of gene pairs in terms of correlation. The improvement in pairwise correlation is measured in terms of SP distance before and after warping. This experiment is performed on the geneset of 39 genes that are present in both the statistical and biological geneset. For each pair of genes, we calculate the correlation between the two matrices, one for each gene. As mentioned earlier in Section 3.1 (Real Dataset), a matrix for a gene represents 11 time series data (for 11 patients) across 6 time points. We first compute the Pearson's correlation on the raw data before we apply the dynamic time warping algorithm. Next, we calculate the pairwise correlation again on the stretched version of data after we apply our algorithm. Figure 5 shows a scatter-plot of the gene correlation values before and after stretching. Along the x-axis we plot the correlation before stretching. Along the y-axis we plot that after stretching. We consider a correlation value between two genes g_i and g_j, to be significant if the absolute value of their correlation is at least 0.4. Our results demonstrate that all the gene pairs (except for one) have a low correlation value before stretching (within ± 0.4). However, after aligning them using our method, the stretched time series of a substantial number of genes yield high correlation

Table 1: The four gene pairs (highlighted in Figure 5 by red dots), having highest absolute correlation value after warp is shown in this table. Each row represents a gene pair information (column A) with correlation without warping (column B), with warping (column C) and shortest path length (column D) in the biological network, retrieved from STRING database.

A	B	C	D
{DZIP1, IGF2BP3}	0.26	0.65	2
{PELO, TAPT1}	0.08	0.63	3
{ITGB1, STK3}	0.29	0.62	2
{DST, MYO1B}	-0.01	-0.60	3

values. Of all the gene pairs, the four gene pairs that have correlation ≥ 0.6 and correlation ≤ -0.6 are highlighted with red dots in Figure 5. Table 3.3 shows the correlation before time warping and after time warping for each of these four pairs. It can be seen that there is a substantial improvement of correlation after applying our algorithm. In order to determine the biological significance of these pairs, we calculate the shortest path length between the two genes in each pair in the human network extracted from STRING database [4]. Table 3.3 shows shortest path length for each gene pair calculated on human network extracted from STRING. The minimum and maximum shortest path length is 2 and 3, respectively. The conjecture is that the less the shortest path distance, the more the biological significance in terms of carrying out a common biological process or molecular function. This signifies that with so less shortest path length, the gene pairs carry out common biological functions.

4 Conclusion

In this paper, we developed a method to align multiple time series gene expression data by using dynamic-time warping approach. Our algorithm first generates an initial distance matrix by calculating pairwise time warping distance between all possible combinations of time series data. It then constructs a guide tree that determines the ordering at which we should align the time series samples, in an iterative manner. The guide tree tends to align the time series pairs with minimum time warping distance first. Next, we build alignment matrix in an iterative manner to align all time series data in an order as determined in the previous step. At a particular iteration of alignment, we align a time series or a group of time series with an already aligned group of time series while preserving all their initial alignments. The alignment is independent of the length of different time series under consideration. Our results on real dataset show that pairwise gene correlation of biologically related genes improve significantly after our method.

References

[1] Technical report: Identifying temporal variation of transcription in populations. `https://www.cise.ufl.edu/~sarkar/MultipleTimeSeriesAlignment_TR.pdf`.

[2] John Aach and George M Church. Aligning gene expression time series with time warping algorithms. *Bioinformatics*, 17(6):495–508, 2001.

[3] Riccardo Bellazzi and Blaž Zupan. Towards knowledge-based gene expression data mining. *Journal of Biomedical Informatics*, 40(6), 2007.

[4] Andrea Franceschini et al. String v9. 1: protein-protein interaction networks, with increased coverage and integration. *Nucleic Acids Research*, 41(D1), 2012.

[5] Yury Goltsev and Dmitri Papatsenko. Time warping of evolutionary distant temporal gene expression data based on noise suppression. *BMC Bioinformatics*, 10(1), 2009.

[6] Christos Hatzis et al. Effects of tissue handling on rna integrity and microarray measurements from resected breast cancers. *Journal of the National Cancer Institute*, 2011.

[7] Kazutaka Katoh, Kazuharu Misawa, Kei-ichi Kuma, and Takashi Miyata. Mafft: a novel method for rapid multiple sequence alignment based on fast fourier transform. *Nucleic acids research*, 30(14):3059–3066, 2002.

[8] Arthur Liberzon et al. Molecular signatures database (msigdb) 3.0. *Bioinformatics*, 27(12), 2011.

[9] Jennifer Listgarten, Radford M Neal, Sam T Roweis, and Andrew Emili. Multiple alignment of continuous time series. In *Advances in neural information processing systems*, pages 817–824, 2005.

[10] Xueli Liu and Hans-Georg Müller. Modes and clustering for time-warped gene expression profile data. *Bioinformatics*, 19(15), 2003.

[11] Cédric Notredame, Desmond G Higgins, and Jaap Heringa. T-coffee: A novel method for fast and accurate multiple sequence alignment. *Journal of molecular biology*, 302(1):205–217, 2000.

[12] Sean Robinson, Garique Glonek, Inge Koch, Mark Thomas, and Christopher Davies. Alignment of time course gene expression data and the classification of developmentally driven genes with hidden markov models. *BMC bioinformatics*, 16(1):196, 2015.

[13] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1), 1978.

[14] David Sankoff and Joseph B Kruskal. Time warps, string edits, and macromolecules: the theory and practice of sequence comparison. *Reading: Addison-Wesley Publication, 1983, edited by Sankoff, David; Kruskal, Joseph B.*, 1983.

[15] Adam A Smith et al. Similarity queries for temporal toxicogenomic expression profiles. *PLoS Computational Biology*, 4(7), 2008.

[16] Adam A Smith et al. Clustered alignments of gene-expression time series data. *Bioinformatics*, 25(12), 2009.

[17] Adi L Tarca et al. Analysis of microarray experiments of gene expression profiling. *American Journal of Obstetrics and Gynecology*, 195(2), 2006.

[18] Julie D Thompson, Desmond G Higgins, and Toby J Gibson. Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic acids research*, 22(22):4673–4680, 1994.

Integrated Metabolic Flux and Omics Analysis of *Leishmania major* metabolism

Sushil Shakyawar[1, 2], Isabel Rocha[1, 2], and Sonia Carneiro[1,*]

[1]SilicoLife Lda, Braga, Portugal
[2]University of Minho, Braga, Portugal
*Corresponding author: scarneiro@silicolife.com

Abstract

Leishmaniasis is a virulent parasitic infection that causes a significant threat to human health worldwide. The existing drugs are becoming less effective due to the ability of *Leishmania spp.* to alter its metabolism to adapt to harsh environments. Understanding how this parasite manipulates its metabolism inside the host (*e.g.* sandfly and human) might underpin new ways to prevent the disease and develop effective treatment strategies.
Despite significant advances in omics technologies, biochemistry of parasites still lacks the understanding of molecular components that determine the metabolic behavior under varying conditions. Metabolic network modeling might be of interest to identify physiologically relevant nodes in a metabolic network.

The present work proposes a metabolic model *i*SK570 (an extension of the *i*AC560 model) with additional reactions for the metabolism of lipids, long chain fatty acids and carbohydrates to study the metabolic behavior of this parasite. Gene Inactivity Moderated by Metabolism and Expression (GIMME) algorithm was used to verify the consistency between model flux predictions and gene expression data. Improved flux distributions were obtained, allowing a more accurate understanding of stage-specific metabolism in of promastigotes and amastigotes.

1. Introduction

Protozoan parasites from the genus *Leishmania* belong to the family Trypanosomatidae, and cause a spectrum of human diseases affecting around 12 million people worldwide (www.who.int). Existing treatment therapies involving drugs such as *e.g.* sodium stibogluconate and meglumine antimoniate, amphotericin B and miltefosine are limited by various features, including in some cases host toxicity and lack of efficacy [1,2]. Considering endemic severity of the disease, there is an urgent need for understanding *Leishmania* metabolism which can subsequently help in developing novel anti-leishmanial therapies.

Significant alterations have been observed in the metabolism exhibited by *Leishmania* at different stages of its life cycle, where it faces different nutritional environments [3]. For example, the promastigote form (inside sandfly) of *Leishmania* preferably uses glucose and L-proline via glycolysis pathways and TCA cycle; while amastigote uses glucosamine (GlcN) and its derivative N-acetylglucosamine (GlcNAc)along with some lipids and amino acids [4,5]. Availability of various sugars, such as hexoses (*e.g.* glucose, mannose, and galactose) and amino sugars (*e.g.* GlcN and GlcNAc) are determining factors for parasitic metabolic phenotype, especially for synthesizing essential glycans and glycoconjugates [6].

Unfortunately, no previous studies have explained the metabolic basis leading to the biosynthesis of glycans and glycoconjugates in the presence of different environments. In fact, it is still unknown if observed metabolic changes are resulting from, or arising out of the different parasitic stages. For example, under promastigote stage, only a few enzymes from the TCA cycle are active, while in amastigote stage, glycolytic enzymes are less functional.

Metabolic network modeling is an effective and sophisticated approach for systematically study the metabolic behaviour of an organism, as well as to understand the relationship between its genotype and phenotype. Previously, these methods have been used to understand the cellular metabolism as well as to identify essential genes in many medically important organisms, such as *Mycobacterium tuberculosis* [7], *Acinetobacter baumanii* [8], *Francisella tularensis* [9] including human parasites like *Leishmania major* [10] and *Plasmodium falciparum* [11]; though with the low prediction accuracy. One of the most probable and obvious reasons for the low prediction accuracy might be associated with the lack of use of experimental data (*e.g.* transcriptomics, proteomics, and metabolomics etc.) to constrain the model and unavailability of the suitable strategies to use omics data in the metabolic network analyses.

Integrating omics data with metabolic network analysis can improve our understanding on various aspects, such as metabolic alterations associated with the environmental conditions, essential genes and metabolic flux variability of the essential reactions [12]. The relevant data can be integrated into the metabolic model to provide an extra layer of metabolic flux constraints to improve its overall prediction efficiency. Various methods like GIMME [13], iMAT [14], MADE [15], E-Flux [15] and PROM [16] have been made available for the integration of transcriptomics and genomics, fluxomics [17], and metabolomics [18] data into metabolic models. Successful examples include the integration of RNAseq data into the *Leishmania infantum* model [19], proteomics data into a metabolic model of *Enterococcus faecalis* [20], and multi-omics data into metabolic models of *Escherichia coli* [21] to understand the metabolism and associated phenotypes. The strategy has also improved drug target predictions in many medically important organisms such as *Aspergillus fumigatus* [22], *Plasmodium falciparum* [23] and *L. major* [24].
In spite of the availability of abundant omics data and various methodologies, only a few studies have employed these strategies to understand the metabolism of *Leishmania* [10,19,25]. Here, we applied omics data with metabolic modeling approaches to understand the metabolic profile of *L.*

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

major under different environmental conditions. The workflow mainly includes the integration of gene expression data from promastigote and amastigote stages into our metabolic model using Gene Inactivity Moderated by Metabolism and Expression (GIMME) method [13].

2. Methodology

2.1. Model extension and refinement

The existing metabolic model *i*AC560 [10] was extended to include sugar nucleotides biosynthetic pathways, which reactions and enzyme-coding genes were collected from databases like KEGG [26] and LeishCyc [27]. As some of the reaction steps were not associated with a specific gene, homology search tools like BLAST [1], were applied to find the highest scoring gene sequences (% identity ≥ 40%, alignment length ≥ 70% and E-value $1.0e^{-30}$), as described in [19] and associate those to the corresponding reactions. Additionally, based on experimental evidence, several metabolic reactions were altered in terms of reversibility and/or compartments, while new transport reactions for sugar nucleotides, lipids, and fatty acids were also included. Refer to **Supplementary material S1** for added, deleted or altered reactions.

2.2. Biomass composition

The macromolecular composition of *L. major* cells was also corrected. Protein, DNA and RNA contents were estimated from *L. donovani* studies [19], while carbohydrates, lipids, and polyamine contents were calculated using experimental data from protozoan *Tetrahymena* [28,29] and *L. mexicana* [30]. Individual carbohydrates, such as mannan, lipophosphoglycan (LPG), glycoinositol phospholipid (GIPL), and N-glycans, were estimated as follows: mannan contents were assumed to represent 80% and 90 % of all carbohydrates in promastigote and amastigote stage, respectively [30], while LPG, GIPL, and N-glycans would represent 20% and 10% in total, respectively. The relative mass fractions (w/w) of LPG, GIPL, and N-glycans were estimated based on previous studies [31–34]. Further details on biomass calculations can be found in **Supplementary material S2.**

2.3. In-silico media formulation

2.3.1. Modified Media for Promastigote (MMP)

MMP was formulated for *L. major* growth under promastigote stage, which includes 16 nutrient sources: L-arginine, L-cysteine, L-histidine, L-isoleucine, L-leucine, L-lysine, L-methionine, L-phenylalanine, L-threonine, L-tyrosine, L-valine, hypoxanthine, phosphate, oxygen, proline, and glucose. The nutrients, in particular, glucose and proline were considered based on the previous studies [35,36], explaining that both the compounds are major carbon source for *Leishmania* promastigote, while remaining ones were included considering the experimental studies [37,38] and computational predictions in [10], which concluded that *Leishmania* can grow in these nutrients.

2.3.2. Modified Media for Amastigote (MMA)

MMA includes all 16 nutrients from MMP with additional amino sugars, amino acids, lipids and fatty acids, making a total of 21 nutrients. GlcN and GlcNAc sugars were added considering findings from Naderer et al. (2010) studies [4] that showed the degradation of glycosaminoglycans inside macrophages to provide GlcN and GlcNAc as carbon sources during the amastigote stage. Fatty acids like stearyl acid and lipids, *e.g.* phosphatidylethanolamine were also considered, based on different studies that show that *Leishmania* utilizes lipids from host cells and transports them into the cytosol [39,40]. The consumption of the lipids and fatty acids during amastigote stages were also supported by other experimental studies discussing the possibility of growth of *Leishmania* axenic amastigote in lipid and fatty acid-rich medium [41–43]. The amino acids aspartate and alanine were also added to MMA, based on higher consumption measurements of these amino acids as carbon sources by amastigotes [44].

2.4. Reaction flux constraints in FBA-based simulations

Model simulations under amastigote and promastigote stages were estimated using different reaction constraints. For example, the uptake flux for proline was reduced by 90% in amastigote compared to promastigote simulations, based on the previous study showing a decrease in the consumption of this particular amino acid in *L. mexicana* amastigotes [35]. Also, glucose uptake flux was constrained to 90% less than that in the promastigote stage, considering previous findings [45,46], which concluded that parasitophorous vacuole is a compartment poor in glucose. Furthermore, the oxygen uptake in amastigote stage was significantly reduced as compared to that in the promastigote stage, considering the fact that *Leishmania*-infected macrophage is an oxygen-deficient entity [47,48]. The upper and lower limits for uptake fluxes for all other nutrients were set unconstrained (See **Supplementary material S2**).

2.5. Metabolic network analysis

The gene expression data (FPKM[1] values) of 10275 genes from *Leishmania* spp. [49] was integrated with the extended metabolic model (termed as *i*SK570) by applying GIMME approach. OptFlux modules [50] were used to run GIMME algorithm and to perform FBA-based analyses under different environmental conditions.

Briefly, GIMME implementation considers genes (and associated reactions) with an expression level below the threshold as inactive, and thus removes those from the simulation. The algorithm may reconsider few of these

[1] FPKM (Fragments Per Kilobase Million) is method for estimating relative abundance of transcripts in terms of fragments observed in RNA-Seq experiment.

inactive reactions, especially the essential ones and so-called metabolically important reactions (MIRs), back in the simulation to achieve an optimal solution. The remaining reactions are blocked and termed as metabolically unwanted reactions (MURs) in that particular metabolic state. Inconsistencies between the metabolic model and gene expression data are estimated based on MIRs that are re-inserted in the model; however, GIMME solves a linear programming (LP) on reconsidered reactions to minimize this inconsistency. As such, inconsistency scores (IS) are calculated and associated with each metabolic reaction. Accordingly, metabolic reactions can be categorized as follow:

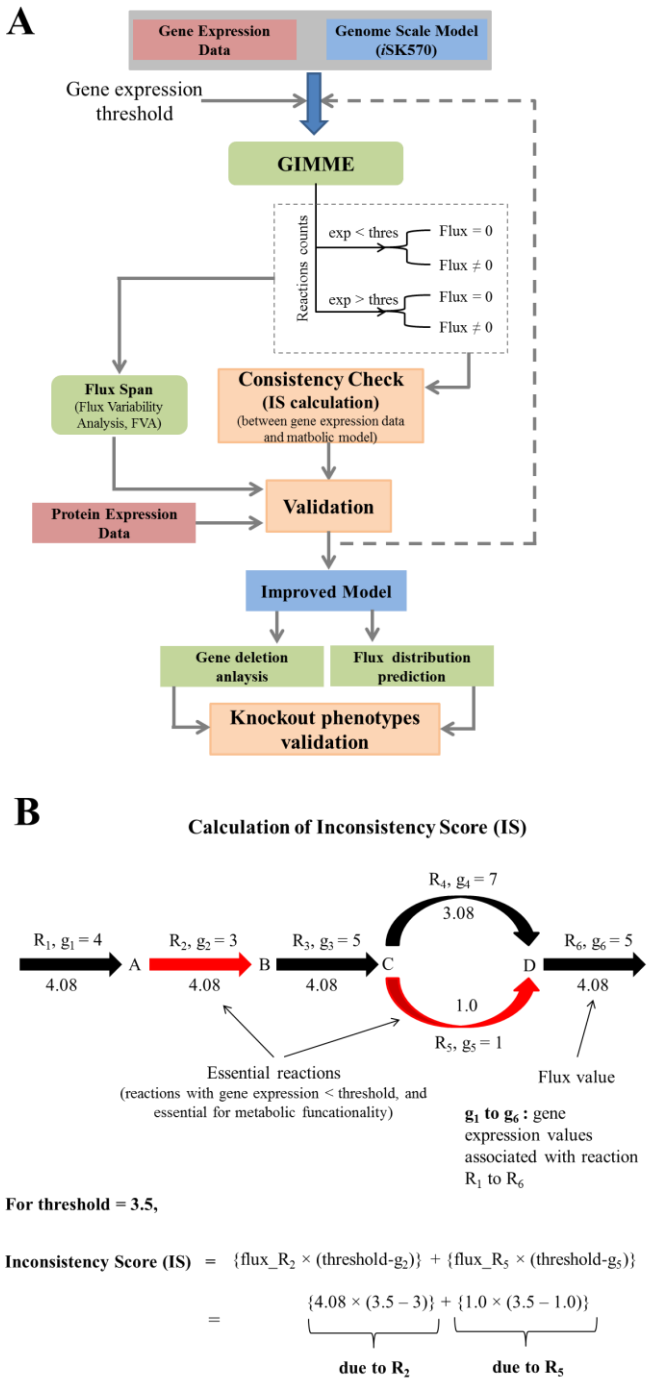


Figure 1: A) Workflow for integrating gene expression data into the metabolic. **B)** An exemplifying scheme for calculating inconsistency score (IS) using gene expression and flux values.

(1) inactive (expression levels below the threshold and metabolic flux[2] equal to zero);
(2) potentially inactive (expression levels below the threshold and metabolic flux[2] is non-zero);
(3) potentially active (expression levels above the threshold and metabolic flux[2] equal to zero);
(4) active (expression levels above the threshold and metabolic flux[2] is non-zero).

Different threshold values were tested, and inconsistency scores (IS) were recalculated as described in [13] (**Figure 1**). Furthermore, flux spans[3] based on Flux Variability Analysis (FVA) and PFBA flux distributions were compared. The predicted changes in metabolic operability of reactions after GIMME implementation were also compared with proteomic data from Pawar et al. (2014) [51].

3. Results and Discussion

3.1. Consistency between metabolic model *i*SK570 and gene expression data in promastigote conditions

Based on different tests, where the gene expression threshold values were changed, it was observed that IS values increase with the threshold values (**Figure 2A**), particularly above threshold values of 11 (**Figure 2B**). Below this threshold, IS values are close to zero, indicating that there are only a few inconsistencies between predicted fluxes and gene expression levels associated to the corresponding reactions. As such, while increasing the threshold value more reactions with predicted fluxes different from zero, but with low expression levels, i.e. reactions that should be active, are included, which increases the level of inconsistency between expression data and flux predictions. Although the number of potentially inactive reactions, i.e. reactions with expression levels below the threshold and predicted zero flux, increases with the threshold value, agreeing with metabolic predictions; the fact is that increasing the threshold value tends to exclude reactions that should be active as predicted by FBA-based simulations.

[2] Metabolic flux was calculated by performing GIMME which uses Parsimonious Flux Balance Analysis (PFBA) to run simulations.
[3] Flux span refers to the difference between maximum and minimum flux values that a reaction can carry according to FVA analysis.

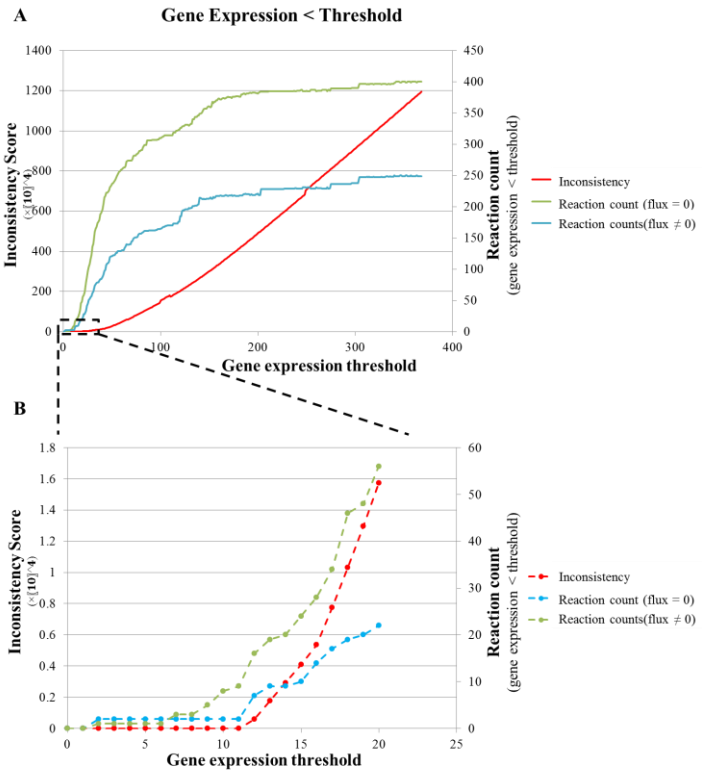


Figure 2: Evaluating inconsistencies between *i*SK570 model predictions and gene expression data from *L. major* promastigote cells. **A)** Inconsistency scores (IS) were calculated for different expression threshold values, while estimating the number of reactions with gene expression levels below a threshold value and predicted flux values equal and different from zero. **B)** Zoom in of plot A for lower threshold values, showing the variation in the inconsistency score and the number of reactions with gene expression below threshold flux values different and equal to zero.

As shown in **Figure 2A,** the number of potentially inactive reactions (i.e. with gene expression less than the threshold and predicted flux equal to zero) increases to a maximum of 400 at the highest expression threshold value (368). In general, GIMME considers these reactions as MURs (or metabolically unwanted reactions) and, ultimately they do not have an impact on the flux distribution. Similarly, potentially active reactions (i.e. with gene expression less than threshold and flux equal to zero) can be associated with MIRs and GIMME might need to reconsider some of these reactions during the simulation process. These are almost 250 at a maximum threshold value of 368. Although the number of MIRs are lower than the number of MURs at a particular threshold value, these contribute far more to increase IS values. Therefore a threshold value should be carefully selected. In the following analysis, a threshold value of 12 (equivalent IS = 5.9×10^{2}) was chosen to perform GIMME simulations, which predicted 30 genes (out of 570) with expression levels below the threshold value, corresponding to 23 reactions from which 16 were considered MURs and 7 MIRs.

3.2. FVA and PFBA analyses

FVA analyses were performed based on GIMME results using a threshold value of 12. Briefly, the idea was to evaluate changes in metabolic predictions imposed by GIMME constraints (especially blocked reactions or MURs) and estimate the impact in the predicted metabolic flexibility under the defined conditions. Therefore, FVA analyses with and without GIMME constraints were compared. Reactions were categorized as such: type1, minimum and maximum FVA fluxes equal to zero; type2, minimum and maximum FVA fluxes different from zero (either positive or negative); and type3, minimum and maximum FVA fluxes equal to upper and lower bounds of reactions (**Table 1**).
Results show that the number of reactions type3 decreased, while reactions type1 and type2 increased, which suggests that GIMME-based constraints reduced metabolic flexibility associated with large FVA spans as defined by FVA fluxes of type 3 reactions. Also, reactions type 1 with FVA spans of zero (i.e. blocked reactions) contribute to decrease this metabolic flexibility, as the number of possible alternatives for carbon distribution within the network also decreases. Minimum and maximum flux values from FVA analyses with and without GIMME constraints for each reaction are presented in **Supplementary material S1.**

Table 1: Number of reactions classified as type1, type2 and type3 from FVA results considering simulations with and without GIMME-based constraints (i.e. deleting MURs).

Reaction Category	Minimum (min) and maximum (max) FVA values	Number of reactions	
		Without GIMME-based constraints	With GIMME-based constraints
type1	min = 0 and max = 0	472	493
type2	min/max< 0 or min/max > 0	239	256
type3	min =lower bound and max=upper bound	466	428

Additionally, PFBA and GIMME flux distributions were compared. In general, flux distributions did not change significantly, most likely because of small differences in the number of active and non-active reactions (**Figure 3A**); however, few reactions changed their flux values from zero to non-zero and vice-versa. The reactions with these binary changes are mostly transport reactions, but reactions associated with metabolic pathways like "Glycerolipid metabolism" (30.3 %) and "Pyrimidine metabolism" (9.09%) (**Figure 3B**) were also found. Changes in flux operability of these reactions can be supported by proteomic data for *L. major* from Pawar et al., 2014 [51], which showed that genes associated with eight reactions (out of ten) that changed their fluxes from zero to non-zero, are expressed at the protein level (**Table 2**). This indicates that GIMME-based flux analyses improve model predictions.

Table 2: List of reactions which showed binary changes (zero to non-zero) in their fluxes after GIMME implementation (threshold value of 12), and which associated enzymes have positive expression at protein level.

Reaction ID	PFBA flux value		Associated genes	Protein expression [51]
	Without GIMME	With GIMME		
R_AGPATi_LM	0	4.80	LmjF32.1960	yes
R_CDPDSPm_LM	0	3.26	LmjF14.1200	yes
R_GPAM_LM	0	4.80	LmjF34.1090	yes
R_HEXg	0	99.30	(LmjF21.0250 or LmjF36.2320) or LmjF21.0240)	yes
R_ME1x	0	162.49	LmjF24.0770	yes
R_PAPAm_LM	0	1.04	(LmjF18.0440 or LmjF19.1350)	yes
R_PNS1	0	10000	LmjF29.2800	yes
R_UPPRTr	0	-10000	LmjF34.1040	yes

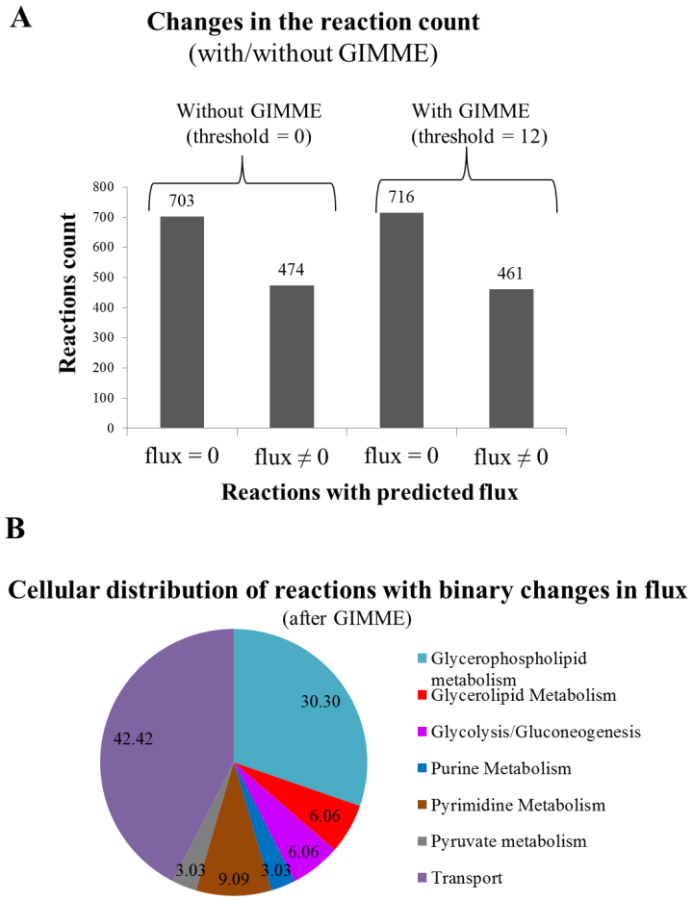


Figure 3: A) Changes in the number of reactions with predicted flux = 0 or ≠ 0 after GIMME implementation. **B)** Percentage cellular distribution of the reactions which showed binary changes in their fluxes after GIMME.

4. Conclusion

The work described the application of GIMME algorithm in combination with flux-based analysis to integrate gene expression data into genome-scale models to determine consistency between data and metabolic model *i*SK570. The strategy has been used to put an extra layer of stoichiometric constraints on reactions to predict more accurate fluxes across various pathways of *L. major*. The predicted activation/inactivation of the metabolic reactions in a particular environment was supported by expression of the associated enzymes at the protein level. Improved flux distribution further used to describe stage-specific metabolism and drug target predictions in *Leishmania* (not described here due to page limitations). All supplementary data mentioned in this manuscript can be provided on demand.

5. Funding and Acknowledgement

This work was supported by the Initial Training Network, GlycoPar, funded by the FP7 Marie Curie Actions of the European Commission (FP7-PEOPLE-2013-ITN-608295). The authors gratefully express appreciation to SilicoLife Lda for providing required infrastructural facilities related to this work. We also thank Bruno Pereira (systems biologist at SilicoLife) and Hugo Giesteira (programmer at SilicoLife) for scientific and technical assistance during various phases of the project.

6. References

1. Philippe J. Guerin, Piero Olliaro, Shyam Sundar, Marleen Boelaert, Simon L. Croft, Philippe Desjeux, Monique K. Wasunna, and Anthony D.M. Bryceson. Visceral leishmaniasis: current status of control, diagnosis, and treatment, and a proposed research and development agenda. *Lancet Infect. Dis*, 2:494–501, 2002.
2. Goto H, Lindoso JA. Current diagnosis and treatment of cutaneous and mucocutaneous leishmaniasis. Expert Rev Anti Infect Ther, 8:419–433, 2010.
3. Fred R. Opperdoes and Graham H. Coombs. Metabolism of Leishmania: proven and predicted. Trends Parasitol, 23:149–158, 2007.
4. Thomas Naderer, Joanne Heng, and Malcolm J. McConville. Evidence that intracellular stages of Leishmania major utilize amino sugars as a major carbon source. PLoS Pathog, 6, 2010.
5. Thomas Naderer, Miriam Ellis, M Fleur Sernee, David P. De Souza, Joan Curtis, Emanuela Handman, and Malcolm J. McConville. Virulence of Leishmania major in macrophages and mice requires the gluconeogenic enzyme fructose-1,6-bisphosphatase. Proc. Natl. Acad. Sci. U. S. A., 103:5502–5507, 2006.
6. Daniel C. Turnock and Michael A. J. Ferguson. Sugar nucleotide pools of Trypanosoma brucei, Trypanosoma cruzi, and Leishmania major. Eukaryot. Cell, 6:1450–1463, 2007.
7. Dany J. V. Beste, Tracy Hooper, Graham Stewart, Bhushan Bonde, Claudio Avignone-Rossa, Michael E. Bushell, Paul Wheeler, Steffen Klamt, Andrzej M. Kierzek, and Johnjoe McFadden. GSMN-TB: a web-based genome-scale network model of Mycobacterium tuberculosis metabolism. Genome Biol., 8:R89, 2007.
8. Hyun U. Kim, Tae Y. Kim, and Sang Y. Lee. Genome-scale metabolic network analysis and drug targeting of multi-drug resistant pathogen Acinetobacter baumannii AYE. Mol. Biosyst., 6:339–48, 2010.
9. Anu Raghunathan, Sookil Shin, and Simon Daefler. Systems approach to investigating host-pathogen interactions in infections with the biothreat agent Francisella. Constraints-based model of Francisella tularensis. BMC Syst. Biol., 4:118, 2010.
10. Arvind K. Chavali, Jeffrey D. Whittemore, James A. Eddy, Kyle T.

Williams, and Jason A. Papin. Systems analysis of metabolism in the pathogenic trypanosomatid Leishmania major. Mol. Syst. Biol., 4:177, 2008.
11. Germán Plata, Tzu Lin Hsiao, Kellen L. Olszewski, Manuel Llinás, and Dennis Vitkup. Reconstruction and flux-balance analysis of the Plasmodium falciparum metabolic network. Mol. Syst. Biol., 6:408, 2010.
12. Rajib Saha, Anupam Chowdhury, and Costas D. Maranas. Recent advances in the reconstruction of metabolic models and integration of omics data. Curr. Opin. Biotechnol., 29:39–45, 2004.
13. Scott A. Becker and Bernhard O. Palsson. Context-specific metabolic networks are consistent with experiments. PLoS Comput. Biol., 4, 2008.
14. Tomer Shlomi, Moran N. Cabili, Markus J. Herrgård, Bernhard Palsson, and Eytan Ruppin. Network-based prediction of human tissue-specific metabolism. Nat. Biotechnol., 26:1003–1010, 2008.
15. Paul A. Jensen and Jason A. Papin. Functional integration of a metabolic network model and expression data without arbitrary thresholding. Bioinformatics, 27:541–547, 2011.
16. Sriram Chandrasekaran and Nathan D. Price. Probabilistic integrative modeling of genome-scale metabolic and regulatory networks in Escherichia coli and Mycobacterium tuberculosis. Proc. Natl. Acad. Sci. U. S. A., 107:17845–50, 2010.
17. Sharon J. Wiback, Radhakrishnan Mahadevan, and Bernhard Palsson. Using Metabolic Flux Data to Further Constrain the Metabolic Solution Space and Predict Internal Flux Patterns: The Escherichia coli Spectrum. Biotechnol. Bioeng., 86:317–331, 2004.
18. Tunahan Cakir, Kiran R. Patil, Zeynep I. Onsan, Kutlu O. Ulgen, Betul Kirdar, and Jens Nielsen. Integration of metabolome data with metabolic networks reveals reporter reactions. Mol. Syst. Biol., 2:50, 2006.
19. Mahesh Sharma, Naeem Shaikh, Shailendra Yadav, Sushma Singh, and Prabha Garg. A systematic reconstruction and constraint-based analysis of Leishmania donovani metabolic network: identification of potential antileishmanial drug targets. Mol. BioSyst., 277:38245–38253, 2017.
20. Ruth Großeholz, Ching-Chiek Koh, Nadine Veith, Tomas Fiedler, Madlen Strauss, Brett Olivier, Ben C. Collins, Olga T. Schubert, Frank Bergmann, Bernd Kreikemeyer, Ruedi Aebersold, and Ursula Kummer. Integrating highly quantitative proteomics and genome-scale metabolic modeling to study pH adaptation in the human pathogen Enterococcus faecalis. npj Syst. Biol. Appl., 2:16017, 2016.
21. Andrew R. Joyce and Bernhard O. Palsson. The model organism as a system: integrating 'omics' data sets. Nat. Rev. Mol. Cell Biol., 7:198–210, 2006.
22. Martin Kaltdorf, Mugdha Srivastava, Shishir K. Gupta, Chunguang Liang, Jasmin Binder, Anna-Maria Dietl, Zohar Meir, Hubertus Haas, Nir Osherov, Sven Krappmann, and Thomas Dandekar. Systematic identification of anti-fungal drug targets by a metabolic network approach. Front. Mol. Biosci., 3:1–19, 2016.
23. Philipp Ludin, Ben Woodcroft, Stuart A. Ralph, and Pascal Mäser. In silico prediction of antimalarial drug target candidates. Int J Parasitol Drugs Drug Resist, 2:191–199, 2012.
24. Arvind K. Chavali, Anna S. Blazier, Jose L. Tlaxca, Paul A. Jensen, Richard D. Pearson, and Jason A. Papin. Metabolic network analysis predicts efficacy of FDA-approved drugs targeting the causative agent of a neglected tropical disease. BMC Syst. Biol., 6:27, 2012.
25. Abhishek Subramanian and Ram R. Sarkar. Revealing the mystery of metabolic adaptations using a genome scale model of Leishmania infantum. Sci. Rep., 7:10262, 2017.
26. Kanehisa M, Goto S. Kyoto Encyclopedia of Genes and Genomes. Nucleic Acids Res., 28:27–30, 2000.
27. Maria A. Doyle, James I. MacRae, David P. De Souza, Eleanor C. Saunders, Malcolm J. McConville, and Vladimir A. Likic. LeishCyc: a biochemical pathways database for Leishmania major. BMC Syst. Biol., 3:57, 2009.
28. Michael A. Gates, Andrew Rogerson, and Jacques Berger. Dry to wet weight biomass conversion constant for Tetrahymena elliotti (Ciliophora, Protozoa). Oecologia, 55:145–148, 1982.
29. P Hellung-Larsen and a P Andersen. Cell volume and dry weight of cultured Tetrahymena. J. Cell Sci., ; 92 (Pt 2):319–24, 1989.
30. Julie E. Ralton, Thomas Naderer, Helena L. Piraino, Tanya A. Bashtannyk, Judy M. Callaghan, and Malcolm J. McConville. Evidence that Intracellular???1-2 Mannan Is a Virulence Factor in Leishmania Parasites. J. Biol. Chem., 278:40757–40763, 2003.
31. Salvatore J. Turco and David L. Sacks. Expression of a stage-specific lipophosphoglycan in Leishmania major amastigotes. Mol. Biochem. Parasitol., 45:91–99, 1991.
32. Malcolm J. McConville and Mike Ferguson. The structure, biosynthesis and function of glycosylated phosphatidylinositols in the parasitic protozoa and higher eukaryotes. Biochem. J., 294:305–324, 1993.
33. Albert Descoteaux and Salvatore J. Turco. The lipophosphoglycan of Leishmania and macrophage protein kinase C. Parasitol. Today, 9:468–471, 1993.
34. John A. Kink and Kwang P. Chang. N-Glycosylation as a biochemical basis for virulence in Leishmania mexicana amazonensis. Mol. Biochem. Parasitol., 27:181–190, 1988.
35. Hart DT, Graham H. Coombs. Leishmania mexicana: Energy metabolism of amastigotes and promastigotes. Exp. Parasitol., 54:397–409, 1982.
36. Petrie M. Rainey and Nicholle Mackenzie. A carbon-13 nuclear magnetic resonance analysis of the products of glucose metabolism in Leishmania pifanoi amastigotes and promastigotes. Mol. Biochem. Parasitol., 45:307–315, 1991.
37. Timothee Merlen, Denis Sereno, Nathalie Brajon, Florence Rostand, and Jean Loup Lemesre. Leishmania spp.: Completely defined medium without serum and macromolecules (CDM/LP) for the continuous in vitro cultivation of infective promastigote forms. Am. J. Trop. Med. Hyg, 60:41–50, 1999.
38. Frederick L. Schuster and James J. Sullivan. Cultivation of clinically significant hemoflagellates. Clin. Microbiol. Rev., 15:374–389, 2002.
39. Thomas Naderer and Malcolm J. McConville. The Leishmania-macrophage interaction: A metabolic perspective. Cell. Microbiol., 10:301–308, 2008.
40. Kai Zhang, Justine M. Pompey, Fong F. Hsu, Phillip Key, Padmavathi Bandhuvula, Julie D. Saba, John Turk, and Stephen M. Beverley. Redirection of sphingolipid metabolism toward de novo synthesis of ethanolamine in Leishmania. EMBO J., 26:1094–104, 2007.
41. Malcolm J. McConville and Blackwell JM. Developmental changes in the glycosylated phosphatidylinositols of Leishmania donovani. Characterization of the promastigote and amastigote glycolipids. J. Biol. Chem., 266:15170–15179, 1991.
42. Winter G, Fuchs M, McConville MJ, et al. Surface antigens of Leishmania mexicana amastigotes: characterization of glycoinositol phospholipids and a macrophage-derived glycosphingolipid. J. Cell Sci., 107:2471–82, 1994.
43. Kai Zhang, Fong F. Hsu, David A. Scott, Roberto Docampo, John Turk, and Stephen M. Beverley. Leishmania salvage and remodelling of host sphingolipids in amastigote survival and acidocalcisome biogenesis. Mol. Microbiol., 55:1566–1578, 2005.
44. Eleanor C. Saunders, William W. Ng, Jennifer M. Chambers, Milica Ng, Thomas Naderer, Jens O. Krömer, Vladimir A. Likic, and Malcolm J. McConville. Isotopomer profiling of Leishmania mexicana promastigotes reveals important roles for succinate fermentation and aspartate uptake in Tricarboxylic Acid Cycle (TCA) anaplerosis,,glutamate synthesis, and growth. J. Biol. Chem., 286:27706–27717, 2011.
45. Henry W. Murray, Jonathan D. Berman, Clive R. Davies, and Nancy G. Saravia. Advances in leishmaniasis. Lancet, 366:1561–1577, 2005.
46. A Garami and T Ilg. The Role of Phosphomannose Isomerase in Leishmania mexicana Glycoconjugate Synthesis and Virulence. J. Biol. Chem., 276:6566–6575, 2001.
47. Alexander Mahnke, Robert J. Meier, Valentin Schatz, Julian Hofmann, Kirstin Castiglione, Ulrike Schleicher, Otto S. Wolfbeis, Christian Bogdan, and Jonathan Jantsch. Hypoxia in Leishmania major skin lesions impairs the NO-dependent leishmanicidal activity of macrophages. J. Invest. Dermatol., 134:2339–2346, 2014.
48. A. Degrossoli, Wagner W. Arrais-Silva, M. C. Colhone, F. R. Gadelha, P. P. Joazeiro and S. Giorgio. The Influence of Low Oxygen on Macrophage Response to Leishmania Infection. Scand. J. Immunol., 74:165–175, 2011.
49. Alberto Rastrojo, Fernando Carrasco-Ramiro, Diana Martín, Antonio Crespillo, Rosa M. Reguera, Begoña Aguado, and Jose M. Requena. The transcriptome of Leishmania major in the axenic promastigote stage: transcript annotation and relative expression levels by RNA-seq. BMC Genomics, 14:223, 2013.
50. Isabel Rocha, Paulo Maia, Pedro Evangelista, Paulo Vilaça, Simão Soares, José P. Pinto, Jens Nielsen, Kiran R. Patil, Eugénio C. Ferreira, and Miguel Rocha. OptFlux: an open-source software platform for in silico metabolic engineering. BMC Syst. Biol., 4:45, 2010.
51. Harsh Pawar, Santosh Renuse, Sweta N. Khobragade, Sandip Chavan, Gajanan Sathe, Praveen Kumar, Kiran N. Mahale, Kalpita Gore, Aditi Kulkarni, Tanwi Dixit, Rajesh Raju, T. S. Keshava Prasad, H. C. Harsha, Milind S. Patole, and Akhilesh Pandey. Neglected Tropical Diseases and Omics Science: Proteogenomics Analysis of the Promastigote Stage of Leishmania major Parasite. OMICS, 18:1–14, 2014.

A Next Generation Sequencing Approach to Analyze Genes Expression in Breast Cancer Stem Cells

Anushree Tripathi , Gautam K. Verma, Krishna Misra*
Department of Applied Sciences, Indian Institute of Information Technology Allahabad (IIITA), Allahabad, India-211015
*Corresponding author: kmisra@iiita.ac.in

Abstract

Despite progressive development in technologies for breast cancer treatment, many challenging issues still persist. Among these challenges, relapse of breast cancer cells was found to be most critical aspect. These cells have tumor-initiating and metastatic potential. The combination of ALDH+ and CD44+/CD24- is widely recognized as potential biomarker for the identification and characterization of breast cancer stem cells. In recent studies, it has been reported that ALDH+ breast cancer cells are 100 fold more tumorigenic that ALDH- cells. High level of tumorigenicity has been observed in breast cancer cells exhibiting positive expression of CD44 and negative expression of CD24. CD44 expression is significantly correlated with Estrogen Receptor-negative (ER-) breast cancer cells. The association between ALDH+ or CD44+/CD24- with Estrogen Receptor-positive (ER+) breast cancer cells in terms of gene expression still remains unresolved. In the present study, the up- and down-regulation of gene expression have been studied that strongly correlate with ALDH+ or CD44+/CD24- expression in ER- and ER+ breast cancer cells by using next-generation sequencing method. For the effective breast cancer therapy, this novel strategy of targeting significant genes can be used as biomarkers to reduce breast cancer stem cells growth that is likely to suppress breast cancer relapse.

Keywords: Cancer relapse, Breast cancer stem cells, Biomarkers, Estrogen receptor, Next Generation Sequencing.

1. Introduction

Breast cancer is found to be leading cause of deaths specifically in women, adversely affecting survival rates. According to World Health Organization (WHO) it has been estimated that breast cancer is the most frequently observed and affects millions all over the world. GLOBOCAN 2012 report has stated that breast cancer ranks second after lung cancer, accounting 15.4% deaths in developed and 14.3% deaths of all cancers in developing countries [1-3]. Despite tremendous progressive development of techniques for reducing death rates caused by breast cancer, there are many unresolved problems. The most critical aspect of breast cancer treatment is the recurrence of cancer cells that leads to metastasis [3]. A small subset of breast cancer cell population has self-renewal and differentiation potential, termed as breast cancer stem cells or breast tumour-initiating cells. These cells cause tumour initiation, tumour propagation and metastasis [4, 5]. Early diagnosis of breast cancer needs targeting of breast cancer stem cells. According to Kai et al., targeting of breast cancer stem cells by combining histone deacetylase and salinomycin can be used for effective inhibition of tumour growth [6-7]. Such targeting can be possible with the development of biomarkers to identify and characterize breast cancer stem cells. Selective and specific inhibition of cancer stem cells can be possible by using molecular markers [8]. Among different biomarkers, combination of Aldehyde dehydrogenases (ALDHs) or CD44 and CD24 are known to be potential markers to diagnose breast cancer stem cells growth. This combination of markers can be used as a prognostic marker for detecting breast cancer recurrence [9]. ALDHs represent a class of nicotinamide-adenine dinucleotide phosphate-positive (NAD(P)+)-dependent enzymes. These are involved in catalyzing oxidation of endogenous including (amino acids, lipids and vitamins) exogenous (drugs and ethanol) aldehyde substrates to their corresponding carboxylic acids. In breast cancer, ALDH acts as a significant predictor of tumour progression by regulating breast cancer stem cells growth [10, 11]. ALDH strongly affects growth of breast cancer stem cells in multiple ways such as gene expression, protein translation

and signalling pathways [12]. Functionally, ALDHs are involved in modulation of retinoic acid signalling pathway to enhance differentiation of breast cancer stem cells, reduction of reactive oxygen species to protect breast cancer stem cells under oxidative stress, chemotherapy resistance to cause breast cancer relapse [13]. ALDHs affect ER negative breast cancer cell lines.

Breast cancer stem cells expressing CD44+/CD24- phenotypes have been used as another potential marker. Sheridan et al., reported that breast cancer stem cells exhibit CD44+/CD24- phenotypes are highly invasive and facilitate metastasis detection. CD44 is a transmembrane glycoprotein, encoded by CD44 gene on chromosome 11 in humans [14]. It acts as a receptor for hyaluronan or hyaluronic acid. CD44 constitutes major component of extracellular matrix and mediates cell-cell and cell-matrix interaction in metastatic growth of breast cancer stem cells. It has been identified as a cancer stem cell marker [15]. CD24 gene on chromosome 6 encodes CD24 which is a small cell surface protein attached to glycosyl-phosphatidyl-inositol. It was obtained in mice as heat stable antigen. CD24 exhibits variety of physiological functions due to high glycosylation activity and plays important role in cell-cell and cell-matrix interactions [16]. Breast cancer can be classified as Estrogen receptor-positive (ER+) and Estrogen receptor-negative (ER-) cancer types. ER+ tumours of breast cancer possess high gene expression of luminal cells and also known as luminal group. It has been estimated that 75% breast cancers are ER+ type. Besides, ER- cells express low level of Estrogen receptor accounting nearly 30% of breast cancer. Due to low expression level of Estrogen receptor, breast cancer stem cells exhibit high level of drug resistance that ultimately results in high rate of breast cancer relapse in ER- cells [17]. Breast cancer cell line i.e., BT-474 has been identified as ER- whereas MDA-MB-231 has shown positive expression for Estrogen receptor [18].

In the present study, gene expression pattern has been observed in ER- and ER+ breast cancer in context of ALDH+/CD44+/CD24- biomarkers in order to predict significant genes as well as their up- and down-regulation in breast cancer stem cells. Genes of breast cancer stem cells were identified from the population of breast cancer cell line by using retrieved data of ALDH+/CD44+/CD24- markers through mapping and alignment methods of Next Generation Sequencing approach. Based on gene finding, network models have been designed to obtain association of significant genes with signalling pathways.

2. Material and methods

2.1. Data retrieval

Three Sequence Read Archive (SRA) data were downloaded from National Centre for Biotechnology Information (NCBI). First data was associated with transcriptional characterization of different states of cancer stem cells in triple negative breast cancer using ALDH+/CD44+/CD24- markers. Second and third data containing regulatory analysis of nuclear receptor signalling in MDA-MB-231 and BT474 breast cancer cell lines were downloaded. Figure 1 represents the overview of proposed methodology.

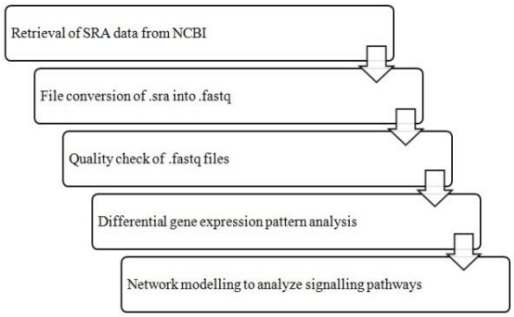


Figure 1: Overview of proposed methodology

Figure 2 depicts the basic steps of proposed methodology. This figure elaborates filtering and mapping tools used in the present study.

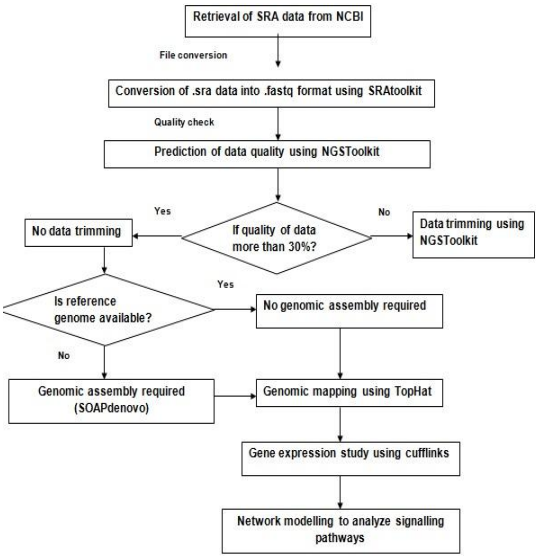


Figure 2: Basic steps of proposed methodology

2.2. Conversion of SRA data into FASTQ

The file format of downloaded .sra files were converted into .fastq format using sratoolkit 2.8. A series of independent data dump utilities constitute SRA Toolkit for the conversion of file formats. SRA Toolkit works on compression by reference using aligned data. This tool stores differences in base pairs between sequence data and the segment it aligns on.

2.3. Quality Control analysis

The quality checking of raw SRA data of three samples were carried out using NGSQC Toolkit 2.3.3. The filtered data for three samples were obtained through quality checking. The NGSQC Toolkit is based on Perl script supported by both Windows and Linux. It provides packages for quality check of sequencing data, trimming of raw data and statistics of quality scores.

2.4. Reference genome retrieval

The reference genome was searched using ensemble database. If the reference genome was available, then the filtered data were used as input for indexing using Bowtie indexer.

2.5. Gene expression analysis with TopHat and Cufflinks

The gene expression analysis of three retrieved data were taken as input for generating index by using Bowtie with DNA sequence in Fasta format. By using TopHat, the alignment of RNA sequence reads to the reference genome was carried out. The specificity of TopHat lies in mapping splice junctions. The transcriptomic assembly from downloaded RNA sequence data was performed with cufflinks. The input file in BAM (Binary equivalent SAM file) format was used to generate assembled isoforms in .gtf format. Cuffmerge module was used to merge transcript assemblies of three retrieved data. Using Cuffdiff, the expression of significant genes were determined.

2.6. Network modelling for predicted genes

Based on predicted genes, the network modelling was carried out using Cytoscape. The network models were prepared to find the connectivity of genes with signalling pathways.

3. Results and Discussion

The significant genes were predicted on the basis of P-value and fold change. The values of P-value and fold change depict up-regulation and down-regulation of genes shown in Table 1 and Table 2. On the basis of fold change of +2 and P-value of 0.05, the up-regulated genes were predicted whereas down-regulated genes were obtained using fold change of -2 and P-value of 0.05.

Table 1: List of significant genes with their P-value and fold change of ER- cancer cell line of MDA-MB-231

S.No.	Gene name	Log_2Fold Change	P-value
1.	PKMP1	10.88	8.10E-06
2.	RP11-500G10.5	9.47	6.37E-05
3.	AL359878.1	9.36	1.64E-05
4.	ANKRD1	9.23	1.59E-05
5.	TMPRSS12	9.63	8.90E-06
6.	ATP5C1P1	14.4	4.68E-06
7.	RP11-64K12.10	8.92	4.16E-05
8.	CORO2B	10.71	3.77E-05
9.	CTB-134H23.2	9.94	2.30E-06
10.	CTC-786C10.1	9.29	1.06E-05
11.	RP11-416I2.1	8.31	7.15E-05
12.	CTD-2231E14.6	9.36	6.99E-06
13.	CTD-2291D10.1	9.02	5.59E-05
14.	CTC-471F3.5	9.93	1.58E-05
15.	AC080125.1	9.10	7.32E-05
16.	ADAMTS6	10.0	8.07E-05
17.	CTC-329D1.2	9.57	6.80E-05
18.	DUSP4	10.1	5.24E-05
19.	RP11-187C18.3	10.8	4.54E-05
20.	RP11-436G20.1	10.8	4.23E-05
21.	RP11-111F5.2	9.91	5.49E-06
22.	AL772307.1	9.94	4.38E-06
23.	RNU6ATAC	9.72	4.19E-06
24.	RNA5-8SP6	12.4	1.37E-05
25.	RP4-561L24.3	12.0	5.06E-08
26.	PRELP	-8.77	2.56E-05

Among twenty five upregulated significant genes shown in Table 1, genes such as ANKRD1, TMPRSS12, CORO2B, ADAMTS6 and DUSP4 were observed to be upregulated and significantly involved in tumorigenesis. In recent studies of Xie et al., ADAMTS6 belongs to a family of A Disintegrin And Metalloproteinase with ThromboSpondin motifs. The dysregulation of ADAMT6 results in the tumor development. The high expression level of ADAMT6 has shown favourable prognosis in breast cancer patients [20].

The up-regulation of ADAMTS6 has been observed in ER- breast cancer cell line. Another gene was found to be up-regulated i.e., TMPRSS12, a transmembrane protease serine 12 protein coding gene possesses endopeptidase activity. ANKRD1gene encodes Ankyrin repeat domain 1 protein. Its expression level is stimulated by the overexpression of p53 and Rac1. Research studies of Kojic et al. [21] reported the peculiar role of ANKRD1 gene as a transcriptional co-activator regulating p53 activity .The functional involvement of ANKRD1 has been observed in monitoring YAP (Yes associated protein). The YAP protein plays key role in tumorigenesis by enhancing self-renewal property and migration of cancer stem cells [22]. In this study, ANKRD1 seems to be involved in ER- breast cancer. Functionally, CORO2B gene encoding actin binding protein i.e, Coronin like protein 2B and mainly contributes in reorganization of actin structure. Previous studies of Toro et al., analyzed the expression of CORO2B in ER+ tumour in context of obesity as one of the risk factor causing breast cancer in postmenopausal women [23]. Its high expression level has been observed in the present work. The Dual specificity phosphatase 4 (DUSP4) gene has significant effect in modulating tumor initiation, mammosphere formation and expression level of $CD44^{+}/CD24^{-}$ markers. It has been reported as negative regulator of MAPK signalling pathway [24]. As shown in Figure 3, the DUSP4 is directly involved in the modulation of MAPK pathway.

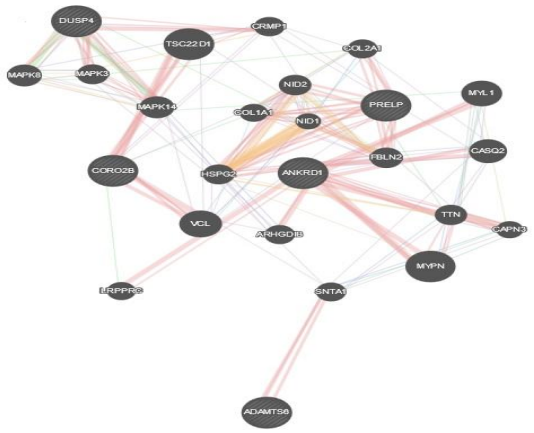


Figure 3: Network model of predicted genes of ER-negative cell line. Physical interactions (pink lines), Predicted genes (Orange), genetic interactions (green), Pathways (cyan).

The proline/arginine rich repeat protein (PRELP) encoded by PRELP gene, expressed in cartilage and bone matrices. It inhibits osteoclastogenesis by reducing NF-κB activity. It represses the growth of breast tumour in causing metastasis through the interaction with microenvironment [25]. The down-regulation of PRELP was observed which needs to be up-regulated for inhibiting breast cancer metastasis.The predicted significant genes in ER+ breast cancer line show different forms of miRNAs, C6orf48 and CD83 as shown in Table 2.

Table 2: List of significant genes based on P-value and fold change of ER+ cancer cell line of BT474.

S.No.	Gene name	Log_2Fold Change	P-value
1.	MIR17 MIR17HG MIR18A MIR19A MIR19B1 MIR20A MIR92A1	10.29	2.03E-06
2.	AC016739.2	10.81	4.52E-06
3.	RNA5SP149	12.79	4.56E-06
4.	C6orf48	13.12	1.81E-07
5.	Metazoa_SRP	8.94	1.23E-05
6.	RNA5-8SP6	10.93	1.50E-07
7.	CD83	-8.33	1.49E-05

Previous studies of Yang et al. [26], revealed the critical role of up-regulated miR-17 in breast tumour progression. The microRNAs are small 20-25 nucleotides and acting as negative regulators of oncogenes. These promote breast carcinogenesis by modulating multiple processes including cellular proliferation, differentiation and metastasis. The finding of present work validates the involvement of miRNA-17 in breast cancer progression. It has been reported that miRNA-18A is closely associated with Dicer dysregulation by suppressing its expression and enhancing Paclitaxel resistance [27]. The high expression level of miRNA-18A has shown its role in breast cancer stem cells in causing drug resistance. The down-regulation of miRNA-20A was observed in breast cancer previously [28] but its up-regulation has been analysed in the present work. CD83 is primarily used as a marker for mature dendritic cells (DC) and also expressed by activated B and T cells. The expression on DC and T cells is essentially important in modulating immune response [29]. Based on studies of Poindexter et al. [30] the low expression level of CD83

was observed in tumour containing sentinel lymph nodes of breast cancer patients. CD83 was found to be down-regulated in breast cancer in this work. Based on prediction of significant genes of ER+ breast cancer cell line using combination of ALDH+/CD44+/CD24- markers, network model was generated to find the connectivity of signalling pathway with predicted genes. Figure 4 shows the network model of ER+ breast cancer cell line.

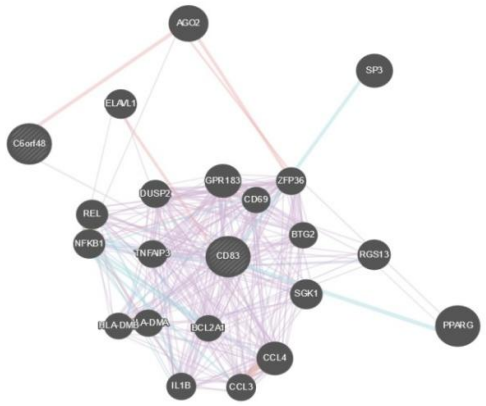


Figure 4: Network model of predicted genes of ER+ breast cancer cell line.

In this model (Figure 4), the NF-κB pathway is associated with predicted significant genes in ER+ cell line of BT474.

4. Conclusion

In order to reduce breast cancer recurrence, the targeting of genes in the small sub-population of cancer stem cells can be considered as important therapeutic strategy. The present work identifies significant genes such as ANKRD1, TMPRSS12, CORO2B, ADAMTS6 and DUSP4 in ER- whereas genes including C6orf48 and CD83 and miRNAs in ER+ breast cancer cell line. These significant genes can be used as potent biomarkers for the identification and characterization of breast cancer stem cells for reducing many obstacles of successful cancer therapy such as tumorigenesis, drug resistance and metastasis. The targeting of signalling pathways of breast cancer stem cells such as MAPK and NF-κB and finding their association with predicted genes can be further exploited for the effective treatment of breast cancer.

Acknowledgement

The author (A.T.) is thankful to the Ministry of Human Resource and Development (MHRD) for fellowship and Indian Institute of Information Technology Allahabad for providing financial support to complete the present work.

References

[1] Carol DeSantis, Jiemin Ma, Leah Bryan, Ahmedin Jemal, Breast cancer statistics, 2013. CA Cancer J. Clin., 64(1): 52-62, 2014.

[2] Ashutosh K. Dubey, Umesh Gupta, Sonal Jain, Breast cancer statistics and prediction methodology: a systematic review and analysis. Asian Pac. J. Cancer Prev., 16(10): 4237-45, 2015.

[3] Xiaomei Ma, Herbert Yu, Global burden of cancer, Yale. J. Biol. Med., 79 (3-4):85-94, 2006.

[4] Thomas W. Owens and Matthew J. Naylor, Breast cancer stem cells, Front Physiol., 4:225, 2013.

[5] Amanda J. Redig and Sandra S. McAllister, Breast cancer as a systemic disease: a view of metastasis. J. Intern. Med., 274(2):113-26, 2013.

[6] Suling Liu and Max S. Wicha, Targeting breast cancer stem cells, J. Clin. Oncol. 28(25): 4006-4012, 2010.

[7] Masaya Kai, Noriko Kanaya, Shang V. Wu, Carlos Mendez, Duc Nguyen, Thehang Luu, Shiuan Chen, Targeting breast cancer stem cells in triple-negative breast cancer using a combination of LHB589 and salinomycin, Breast Cancer Res. Treat., 151(2):281-94, 2015.

[8] Tobias Schatton, Natasha Y. Frank, Markus H. Frank, Identification and targeting of cancer stem cells. Bioessays 31(10): 1038-49, 2009.

[9] Yoshiya Horimoto, Atsushi Arakawa, Noriko Sasahara, Masahiko Tanabe, Sei Sai, Takanori Himuro, Mitsue Saito, Combination of cancer stem cell markers CD44 and CD24 is superior to ALDH1 as a prognostic indicator in breast cancer patients with distant metastases. PLoS One, 11(10): e0165253, 2016.

[10] David W. Clark, Komaraiah Palle, Aldehyde dehydrogenases in cancer stem cells: potential as therapeutic targets, Ann. Transl. Med., 4(24): 518, 2016.

[11] Xia Xu, Shoujie Chai, Pingli Wang, Chenchen Zhang, Yiming Yang, Ying Yang, Kai Wang, Aldehyde dehydrogenases and cancer stem cells. Cancer Lett., 369 (1): 50-7, 2015.

[12] Azam Bozorgi, Mozafar Khazaei, Mohammad Rasool Khazaei, New findings on breast cancer stem cells: A review. J Breast Cancer 18(4): 303-312, 2015.

[13] Hiroyuki Tomita, Kaori Tanaka, Takuji Tanaka, Akira Hara, Aldehyde dehydrogenase 1A1 in stem cells and cancer. Oncotarget 7(10): 11018-11032, 2016.

[14] Carol Sheridan, Hiromitsu Kishimoto, Robyn K. Fuchs, Sanjana Mehrotra, Poornima Bhat-Nakshatri, Charles H. Turner, Robert Goulet Jr, Sunil Badve, Harikrishna Nakshatri, CD44+/ CD24- breast cancer cells exhibit enhanced invasive properties: an early step necessary for metastasis. Breast Cancer Res., 8(5): R59, 2006.

[15] Yongmin Yan, Xiangsheng Zuo, Daoyan Wei, Concise review: Emerging role of CD44 in cancer stem cells: A promising biomarker and therapeutic target. Stem Cells Transl. Med., 4(9): 1033-1043, 2015.

[16] Appalaraju Jaggupilli and Eyad Elkord, Significance of CD44 and CD24 as cancer stem cell markers: An enduring ambiguity. Clin. Dev. Immunol. 2012(2012):708036, 2012.

[17] Ozlem Yersal and Sabri Barutca, Biological subtypes of breast cancer: Prognostic and therapeutic implications. World J. Clin. Oncol., 5(3): 412-24, 2014.

[18] Kristina Subik, Jin-Feng Lee, Laurie Baxter, Tamera Strzepek, Dawn Costello, Patti Crowley, Lianping Xing, Mien-Chie Hung, Thomas Bonfiglio, David G. Hicks, Ping Tang, The expression patterns of ER, PR, HER2, CK5/6, EGFR, Ki-67 and AR by immunohistochemical analysis in breast cancer cell lines. Breast Cancer (Auckl.), 4: 35-41, 2010.

[19] Ravi K. Patel and Mukesh Jain, NGS QC Toolkit: A toolkit for quality control of next generation sequencing data. PLoS One 7(2): e30619, 2012.

[20] Yuxin Xie, Qiheng Gou, Kegi Xie, Zhu Wang, Yanping Wang, Hong Zheng. ADAMTS6 suppresses tumor progression via the ERK signaling pathway and serves as a prognostic marker in human breast cancer. Oncotarget 7(38): 61273- 61283, 2016.

[21] Snezana Kojic, Aleksandra Nestorovic, Ljiljana Rakicevic, Anna Belgrano, Marjia Stankovic, Aleksandra Divac, Georgine Faulkner. A novel role for cardiac ankyrin repeat protein Ankrd1/ CARP as a co-activator of the p53 tumor suppressor protein. Arch. Biochem. Biophy. 502(1), 60-67, 2010.

[22] Giovanni Sorrentino, Naomi Ruggeri, Alessandro Zannini, Eleonora Ingallina, Rebecca Bertolio, Carolina Marotta, Carmelo Neri, Elisa Cappuzzello, Mattia Forcato, Antonio Rosato, Miguel Mano, Silvio Bicciato, Giannino Del Sal. Glucocorticoid receptor signalling activates YAP in breast cancer. Nat. Commun. 8:14073, 2017.

[23] Allyson L. Toro, Nicholas S. Costantino, Craig D. Shriver, Darell L. Ellsworth, Rachel E. Ellsworth. Effect of obesity on molecular characteristics of invasive breast tumors: gene expression analysis in a large cohort of female patients. BMC Obes. 3:22, 2016.

[24] Justin M. Balko, Luis J. Schwarz, Neil E. Bhola, Richard Kurupi, Philip Owens, Todd W. Miller, Henry Gomez, Rebecca S. Cook, Carlos L. Arteaga. Activation of MAPK pathways due to DUSP4 loss promotes cancer stem cell-like phenotypes in basal-like breast cancer. Cancer Res. 73(20):6346-58, 2013.

[25] Nadia Rucci, Mattia Capulli, Luca Ventura, Adriano Angelucci, Barbara Peruzzi, Viveka Tillgren, Murizio Muraca, Dick Heinegard, Anna Teti. Proline/ arginine-rich end leucine-rich repeat protein N-terminus is a novel osteoclast antagonist that counteracts bone loss. J Bone Miner Res. 28(9):1912-24, 2013.

[26] Fangliang Yang, Yuan Li, Lingyun Xu, Yulan Zhu, Haiyan Gao, Lin Zhen, Lin Fang. miR-17 as a diagnostic biomarker regulates cell proliferation in breast cancer. Onco Targets Ther. 10: 543-550, 2017.

[27] L.-Y. Sha, Y. Zhang, W. Wang, X. Sui, S.-K. Lui, T. Wang, H. Zhang. MiR-18a upregulation decreases Dicer expression and confers paclitaxel resistance in triple negative breast cancer. Eur. Rev. Med. Pharmacol. Sci. 20(11): 2201-8, 2016.

[28] Ming-Qi Fan, Chi-Bing Huang, Yan Gu, Ya Xiao, Jin-Xin Sheng, Lin Zhong. Decrease expression of microRNA-20a promotes cancer cell proliferation and predicts poor survival of hepatocelullar carcinoma. J. Exp. Clin. Cancer Res., 32:21, 2013.

[29] Cindy Aerts-Toegaert, Carlo Heirman, Sandra Tuyaerts, Jurgen Corthals, Joeri L. Aerts, Aude Bonehill, Kris Thielemans, Karine Breckpot. CD83 expression on dendritic cells and T cells: Correlation with effective immune responses. Eur. J. Immunol., 37:686-95, 2007.

[30] Nancy J. Poindexter, Aysegul Sahin, Kelly K. Hunt, Elizabeth A. Grimm. Analysis of dendritic cells in tumor-free and tumor-containing sentinel lymph nodes from patients with breast cancer. Breast Cancer Research 6(4): R408-R415, 2004.

Machine Learning and Sentiment Analysis: Examining the Contextual Polarity of Public Sentiment on Malaria Disease in Social Networks

Jelili Oyelade[1, 3, *], Efosa Uwoghiren[1, 3],Itunuoluwa Isewon[1, 3], Olufunke Oladipupo[1], Olufemi Aromolaran[1, 3]& Kingsley Michael[2]

[1]Department of Computer & Information Science, Covenant University, Ota
[2]Department of Accounting, Covenant University, Ota
[3]Covenant University Bioinformatics Research Cluster (CUBRe), Ota

Abstract

Malaria, a major deadly disease which is still a threat to human life's even though numerous efforts has been put to fight it, still affects over two hundred million people each year amongst which over a million individuals dies. Twitter happens to be an important and comprehensive source of information that is quite subjective to individual sentiments towards public health care. In this study, we extracted tweets from the social network twitter, we pre-processed the tweets extracted and built a model to fit our data using a machine learning approach for text classification to determine the contextual polarity of every tweet on the subject of malaria in the bid to harvest peoples' opinion towards malaria and understand how well research and recent development in the aid to tackle malaria has affected the opinions of the public towards the subject malaria. This study finds that tweets extracted, pre-processed and classified in this study were majorly classified as negative (-ve) due to the fact that tweets tweeted were majorly about different occurrence of death, misinformation and need for donations to save a life, hence a major awareness is needed.

Keywords: Sentiment Analysis, Machine Learning Technique, Malaria, Twitter, Data Mining.

1. Introduction

The dominance of malaria resistance to all identified anti-malarial drugs in current circulation has given rise to the increase of anti-malarial drug discovery research [1]–[3]. Hence, research towards the development of novel drug which would serve as effective solutions for malaria treatment are urgently needed [2]–[5] because despite the colossal efforts put in to fight malaria, the disease still affects up to over 200 million people every year amongst which close to half a million dies [1], [5]–[9]. Considering the declaration "Action and Investment to defeat Malaria 2016–2030 (AIM) – for a malaria-free world" [10], we have decided to examine public opinion towards the subject malaria. Opinions are central to almost all human activities because they are key influencers of our behaviours. As humans we love to find out what the public feels or think about a particular brand, topic or subject area this does not leave out finding out what people think about a particular disease and how research and recent developments are affecting the public opinion. With the proliferation of Web to applications such as micro-blogging, forums and social networks, reviews, comments, recommendations, ratings and feedbacks have been made very easy as users can generate content about virtually anything.Twitter, which is a micro blogging platform permits the fast exchange of personal ideas and thoughts, therefore allowing users to tweet messages of about 140 characters [11]–[13]. With the explosion of this user generated content, came the need for companies, service providers, social psychologists, analysts and researchers to mine and analyze these contents for different relevant uses, this is quite significant bearing in mind that tweets are often treated as facts and are cited in information outlets for example news media [14]. The research community and organizations are not left out in the need to find out how their research and recent developments are affecting the public opinion towards their discoveries and novel implementations in a particular subject area, hence, the reason for sentiment analysis. In this study, we have decided to examine if the ground-breaking research and recent developments on the subject malaria have really affected people's opinion and awareness on the subject malaria.

1.1 Natural Language Processing

Natural languages are those languages spoken or written by humans for the purposes of communication. Natural Language processing (NLP) can be defined as "a theoretically motivated range of computational techniques for analysing and representing naturally occurring texts at one or more levels, for the purpose of achieving human-like language processing of tasks and applications"[15], [16]. The field of NLP involves making computers to perform useful tasks with the natural languages for human use. The input and output of an NLP system can be either Speech or Written Text. Below are the different stages in natural language processing.

978-1-943436-11-8 / copyright ISCA, BICOB 2018
March 19-21, 2018, Las Vegas, Nevada, USA

Phonology: This deals with organizing sound systematically. Phonetics and phonology deal with the articulatory and acoustic properties of speech sounds, how they are produced, and how they are perceived, and the rules that govern them [17], [18].

Lexical Analysis:identifying and analysing the structure of words. Lexicon of a language means the collection of words and phrases in a language [19], [20].

Morphological Analysis:Thisrefers to the study of construction of words from primitive meaningful units. Since the meaning of each morphemes are the same across words human can break down an unknown word into constituent's morphemes in order to understand its meaning [21], [22].

Syntactic Analysis: This involves determining the structural role of words in the sentence and in phrases. The words are transformed into structures that show how the words are related to each other. This requires the grammar and the parser. The output of this level of processing is a representation of the sentence that reveals the structural dependencies and relationships between the words [17], [18], [22].

Semantic Analysis:It is concerned with the meaning of words and how to combine words into meaningful phrases and sentences. It assigns meanings to natural language utterances. A semantic representation must be precise and unambiguous. It draws the exact meaning or the dictionary meaning from the text. The text is checked for meaningfulness. It is done by mapping syntactic structures and objects in the task domain [15], [20], [22].

Discourse Analysis: It deals with how the immediately preceding sentence can affect the interpretation of the next sentence. For example the word "it" in the sentence "she wanted it" depends upon the prior discourse context. The meaning of any sentence depends upon the meaning of the sentence just before it. In addition, it also brings about the meaning of immediately succeeding sentences [15], [17], [20], [21].

1.2 Text Mining

Text mining, also known as Intelligent Text Analysis or Knowledge-Discovery in Text (KDT), refers generally to the process of extracting interesting and non-trivial information and knowledge from unstructured text [23]–[26]. Text mining is now widely being applied to many domains, some of its application areas include: Sentiment analysis, Educational application, Security applications, biomedical applications, Digital humanities and Computational sociology etc.

1.3 Sentiment Analysis

Sentiment analysis gives room of harvesting opinions from reviews or expression of different users on a particular subject matter or product. This groups opinions into either negative, positive or neutral helping to determine the attitude or opinion of a particular writer or speaker with respect to certain topics [27]–[29].Sentiment Analysis is considered a classification process. 3 major classification levels makes up sentiment analysis which are the Sentence-level whose its objective is classifying sentiment expressed in subjective sentences as either negative or positive sentiments, the Document-Level whose its objective is classifying sentiment of the whole document as expressing as either negative or positive sentiments and the Aspect Level whose its objective is classifying sentiment with respect to the specific aspects of entities [30]–[35].

1.3.1 Applications of Sentiment Analysis

Sentiment analysis has been applied to various real world scenarios and a few has been considered in this study.

Reputation Monitoring: sentiment analysis has been applied to monitor the reputation of different brands on Facebook and twitter as they are the core of sentiment analysis in this domain [36]

Product and Service Reviews: sentiment analysis has been applied to the reviews of consumer products and services using websites that provide automated summaries of reviews about products and about their specific aspects [27], [37]

Result Prediction: sentiment analysis has been applied to predict the probable outcome of a particular event [38]. For instance, sentiment analysis can provide substantial value to candidates running for various positions enabling campaign track how voters feel about different issues and how they relate to the speeches and actions of the candidates [38]

Decision Making: sentiment analysis has been applied to help decision making process [39]. There are numerous news items, articles, blogs, and tweets about each public company. A sentiment analysis system can use these various sources, articles that discuss the companies and aggregate the sentiment about them as a single score that can be used by an automated trading system [29], [39]

1.3.2 Sentiment Classification

The major aim of sentiment classification is classifying documents or reviews into a definite number of predefined categories. The expressions of opinions in a sentence, document etc. could either be done using a scaling system or binaries (negative and positive).

Feature Selection in Sentiment Classification: Feature selection gives a better understanding of data by giving

their important features. Feature selection is an important step in text categorization problems; not all the features in a document are required to classify it. Feature selection helps in removing unimportant or redundant words of text and thereby reduces the dimensionality of documents [29], [40].

1.3.3 Sentiment Classification Techniques

It can be roughly divided into machine learning approach and lexicon based approach, the lexical based approach relies on a sentiment lexicon, a collection of known and precompiled sentiment terms.

Lexicon-based approach: Opinion words are employed in many sentiment classification tasks. Positive opinion words are used to express some desired states, while negative opinion words are used to express some undesired states. There are also opinion phrases and idioms which together are called opinion lexicon [35], [38], [41], [42].

Machine learning (ML) Approach: Machine learning techniques first trains the algorithm with a training data set before applying it to the actual data set [43]. The text classification methods using machine learning approach can be roughly divided into supervised and unsupervised learning methods [43], [44].

2. Materials and Method

2.1 Phases of the Methodology

Represented in table 1 are the method and techniques used in different phases of this study.

Table 1: Method and techniques used in different phases

Phase	**Methods | techniques**	**Output**
Tweets Collection	Obtaining tweets from Streaming twitter API	Samples of relevant tweets.
Linguistic analysis	Lemmatization, stop words removal, harsh tags removal	Pre-processed tweets
Feature Extraction	Extracting feature vectors to identify sentiment polarity	Feature vectors
Sentiment Analysis	Implementation of the model to classify and identify sentiment polarity	Sentiment polarity, positive, Negative and Neutral

2.2 Design Considerations for the Sentiment Analysis Model

The approach for the development of our sentiment analysis model which involves tweets preprocessing, feature extraction, Machine learning text clustering and classification techniques is being described.

Tweets used were obtained through the Twitter streaming API using python. These sets of tweets serve as the input data for our model. These tweets were preprocessed by removing stop words and harsh tags. Thereafter, lemmatization, POS tagging, were performed to enable the efficient extraction of features for the classification of sentiments of the tweets. After these tweets have been preprocessed, feature vectors were extracted and these vectors were analyzed using machine learning algorithms of clustering and classification techniques, the algorithms to be used include Naïve Bayes and Support vector Machine algorithms.

2.3 System Architecture

This shows the formal description and representation in a way that supports reasoning about the structure and behaviour of the system. It comprises of the individual components and the way they work together to implement the sentiment analysis model. Figure 1 shows the System architecture and its various components.

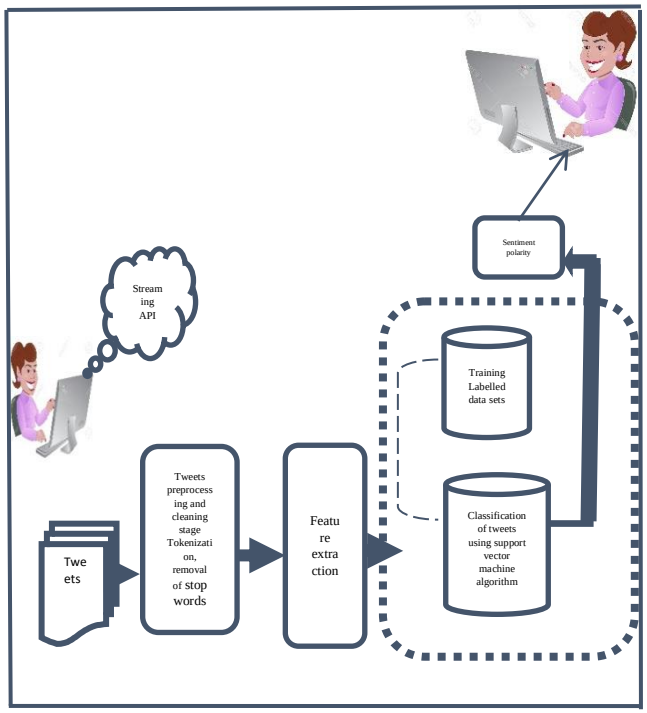


Figure 1: The system architecture of our study

2.4 Framework of the Sentiment Analysis Model

Here the different stages involved in the development of this study sentiment analysis model are comprehensively outlined. It explains the process and components for analysis, design and implementation of the model. Figure 2 shows the entire life cycle of this model which is in 4 major phases namely Tweet Pre-processing, Feature Extraction, Semantic Analysis, and Machine learning Classification.

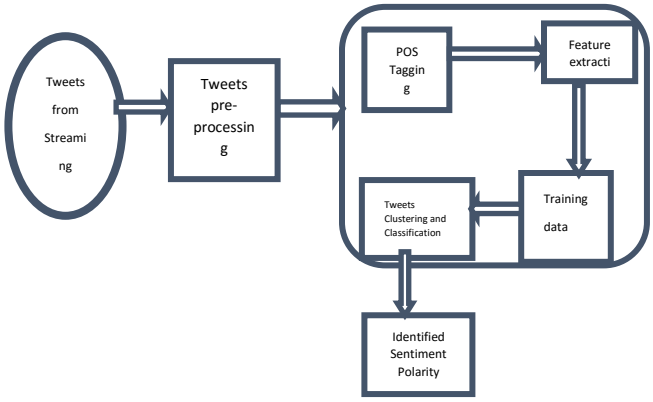


Figure 2: Framework of the Sentiment Analysis Model

2.4.1 Tweets Preprocessing

This is the first stage of the development process. Natural language text can't be processed directly; it must be pre-processed first in order to obtain accurate results at the end of this project. In light of that, the Pre-processing phase includes Tokenization, Language detection, Stop words removal. These processes would be explicitly explained below with corresponding examples.

Tokenization: We tokenized the text to make it easy to separate out other unnecessary symbols and punctuations, and leave out only those words that can add value to the sentimental polarity score of the text. For a sample input text says "Tunde said the food he bought is bad". Tokenizing divides the strings into lists of substrings known as the tokens.

Language detection: This enables only the detection of tweets in English since we are mainly interested in English text only. This is possible by using NLTK's language detection feature.

Stop words Removal: In this process, we removed very common words such as "all", "almost", "alone", etc. The reason for this is because their appearance in a tweet does not provide any useful information in classifying a tweet as positive, negative or neutral.

2.4.2 Part of speech (POS) tagging

POS tagging assigns a tag to each word in a text and classifies a word to a specific morphological category such as noun, verb, adjective, etc. POS taggers are efficient for explicit feature extraction in terms of accuracy.

2.4.3 Feature Extraction

The improved dataset after pre-processing has a lot of distinctive properties. The feature extraction method, extracts the adjective from the dataset. Later this adjective would be used to show the positive and negative polarity in a sentence which is useful for determining the opinion of individuals. This would be done using a Unigram model. Unigram model extracts the adjective and segregates it. It discards the preceding and successive word occurring with the adjective in the sentences. For above example, i.e. "painting ugly" through unigram model, only ugly is extracted from the sentence.

2.4.4 Tweets Classification and Identification of Sentiment Polarity

When using a machine learning approach, the ways features are selected are very important to the success rate of the classification. In this study, we used 2 major machine learning algorithms which are Support Vector Machine (SVM) Algorithm and Naïve Bayes (NB) Algorithm. SVM & NB algorithms are trained using the features generated which results into the training model.

3. Results

3.1 Tweets Collection

Twitter Application Programming Interface (API) which was implemented in python language was used to collect English-language tweets relating to malaria over a period of several weeks. The corpus consists of thousands of tweets from search tags like 'malaria', 'malaria Africa', 'malaria Mozambique', 'malaria Nigeria', etc. This task was performed by different experts.

Table 2: Sample of tweets collected

Sample of tweets collected	***Sample of tweets collected***
@Y1079FM @iambrownberry @deejayLoft @LukmanEvergreenHelo brown berry u make my sunday morning amazing wooow.cant s\xe2\x80\xa6 https://t.co/o2x1DPUT7z Currently sat under my	*@SandeshRishi @altnews_in @mepratap @timesofindia You are so right, the girl died because of Malaria !\xe2\x80\xa6 https://t.co/CibEtxBtk7 Do you know that hunger kills more people annually than AIDS, malaria and*

mosquito. net listening to TMS in Nigeria suffering with malaria! #bbccricket #TMS *RT @fulelo: Extreme gardening to help tackle malaria https://t.co/ojaXL51W6J* *RT @Beanchesterr: Latest research has it that bank alerts can cure malaria.* *Extreme gardening to help tackle malaria https://t.co/ojaXL51W6J* *@GRadioRockstar Lmao and the you read through and find out "Big Daddy Juix was exposed to malaria" damn headlines*	*tuberculosis combined?\xe2\x80\xa6 https://t.co/51bh4ItiiD* *Is there a way to threat Malaria without taking pills or injections???* *RT @scienmag: Chances of surviving malaria may be higher when host consumes fewer calories https://t.co/NCF8WH7JFP https://t.co/AAxOaotUz3* *@Blackkout__ I got very sick at they little age... I was down with malaria 3+.*
I was very OK but the stress was too much. And it's not like I did any tedious work. Just moving about caused malaria. *RT @ISGLOBALorg: Beatriz Galatas @beagalatas: Working on #malaria elimination in #Mozambique https://t.co/esVX5jrSmX @Manhica_CISM @FundlaC\xe2\x80\xa6* *After taking malaria meds i always think to myself "dont go to sleep, dont go to sleep..." https://t.co/egUI8rPWWe* *RT @PaulUithol: Second day of @sotmafrica! Kicking off with*	*Before he was diagnose of malaria in during the week and couldn't make it by weekend.So sad. I wish the Olajide's fortitude to bear the loss* *RT @PreventionTips: Malaria, Mosquitoes and Man: Prevention & Control https://t.co/9Ingffq6xk https://t.co/eMEM1cxZCP* *Expert wants increased awareness on malaria to save more\xc2\xa0lives https://t.co/0jl4tNTWsv https://t.co/iWnEODQh8y* *Don't forget to take your #Malaria pill!*

3.2 Linguistic analysis

The tweets are pre-processed removing stop words, hash tags, duplications and languages that are not known to be English language hereby leaving us with the tweets needed for feature extraction. Below is a sample of the pre-processed tweets.

Table 3: Pre-processed tweets

Pre-processed tweets	*Pre-processed tweets*
extreme gardening to help tackle malaria *latest research has it that bank alerts can cure malaria* *after taking malaria meds i always think to myself dont go to sleep, dont go to sleep* *researchers find that simple blood test predicts anemia risk after malaria treatment* *My barber's friend died last Friday fine boy, waiting for nysc. everything smooth he died of malaria*	*olajide's fortitude to bear the loss* *wants increased awareness on malaria to save more* *2tweetaboutit Africa is the epicentre of malaria. soon there will be an malaria epidemic in Europe*
a human trial for a malaria vaccine has achieved up to 100% protection against infection for at least 10 weeks *mapbox using mapping and visualizations to fight malaria* *is there a way to threat malaria without taking pills or injections ??* *chances of surviving malaria may be higher when host consumes fewer calories* *baygon is almost 2k which is hilarious bc it's technically cheaper for me to get malaria and treat it.*	*world's first malaria vaccine will be given to thousands of babies in Africa* *Heavy rainfall + poor drainage = mosquitoes having the time of their life spreading malaria.* *indigenous knowledge systems and innovations in malaria control in Nigeria* *No wonder Nigeria is still fighting malaria. everywhere waterlogged as fuck* *Mosquitoes in Nigeria will give you malaria. Mosquitoes in Philippines will give you dengue fever, which is very deadly.*
malaria control in African schools dramatically cuts infection and reduces risk of anaemia *u.s. malaria donations saved almost 2m African children* *i got very sick at they little age..i was down with malaria* *you can help us save lives by giving the gift of malaria treatment, which quickly restores*	*antimalarials of unproven quality rampant in africa - sub-saharanafrica* *king mswati ii of Swaziland calls for increased domestic investments to eliminate malaria in Africa* *did you know: in Nigeria it is cheaper to get malaria and treat it than to buy insecticide* *touching needs donations to save children in Nigeria from malaria-please donate*

3.3 Feature Extraction

Below is an improved dataset after extracting the adjective from the pre-processed dataset which becomes useful when classifying the polarities into negative, neutral or positive polarities.

Table 4: Feature vectors

Sample feature vectors	Sample feature vectors
extreme gardening help	*diagnose malaria week make*

tackle malaria	*sad wish fortitude bear loss*
taking malaria meds think myself dont sleep dont sleep	*got sick little age malaria*
friend died Friday fine boy waiting nysc smooth died malaria	*calls strong regulation local production antimalarials*
malaria donations saved African children	*defeat malaria Africa*
chances surviving malaria higher host consumes fewer calories	*malaria vaccine given thousands babies Africa*
baygon hilarious bc technically cheaper malaria treat	*malaria death rate Africa fell*
	Gambia massive progress malaria elimination sight

3.4 Classifier Performance

Our classifier labelled tweet sentiment with an accuracy of about 72.41%. Importantly, no positively classified tweets were manually labelled as negative, and only 2% of the negatively classified tweets were manually labelled as positive by assigning a positive polarity to them. The misclassifications were predominantly for tweets with non-neutral sentiment classified as being neutral. As such, the overwhelming majority of misclassified tweets did not entail complete reversal of sentiment. Below are the accuracy of the five classifiers that were combined and used in this study.

Table 5b: Combinations of Naïve Bayes Classifiers

CombinationsofNaiveBayesClassifiers, fordatasetstraining	
Original Naive Bayes Algo accuracy percent	74.54819277108435
MNB_classifier accuracy percent	73.94578313253012
BernoulliNB_classifier accuracy percent	73.64457831325302

Table 5c: Combinations of Support Vector Machine Classifiers

CombinationsofSupportVectorMachineClassifiers, fordatasetstrainin	
LinearSVC_classifier accuracy percent	70.03012048192771
NuSVC_classifier accuracy percent	69.879518072289

3.4.1 F1-Score

The table 5a below shows the performance analysis of the corpus used for training the classifiers.

Precision Score = tp / (tp + fp)

Recall Score = tp / (tp + fn)

F1-Score: F1 = 2 * (precision * recall) / (precision + recall)

Where: tp = true positives, fp = false positives, fn= false negatives.

Table 5a: F1-score

	Precision	**recall**	**f1-score**	**support**
pos	0.87	0.86	0.87	1600
Neg	0.83	0.89	0.87	1600
Avg / total	0.85	0.88	0.87	3200

3.5 Sentiment Analysis

Every tweet after feature extraction was done separately on them after the model has been trainedwere assigned polarities based on the how it was classified by the model ranging from neutral (0.25, 0.5, 0.75, 1), to negative (0.25, 0.5, 0.75, 1), to positive (0.25, 0.5, 0.75, 1) respectfully. Below are samples of polarities assigned to tweets and a graph diagrammatically representing the polarities of tweets of a period of time?

Table 6: Polarity of tweets extracted

Sample of tweets polarity	**Sample of tweets polarity**
taking malaria meds think myself dont sleep dont sleep |neg|1.0	*diagnose malaria week make sad wish fortitude bear loss |neg|1.0*
friend died friday fine boy waiting nysc smooth died malaria |neg|1.0	*got sick little age malaria |neg|1.0*
baygon hilarious bc technically cheaper malaria treat |neg|1.0	*leaders adopt new strategic framework end aids tb malaria |pos|1.0*
American donations fight malaria Africa saved lives nearly million |pos|0.75	*antimalarials unproven quality rampant Africa scidevnet report malaria |neg|0.75*
people die malaria yearly reduce clean environment join movement |neg|1.0	*Gambia massive progress malaria elimination sight |neu|1*
children world die malaria |neg|0.75	*heavy rainfall poor drainage mosquitoes having time life spreading malaria |neg|1.0*

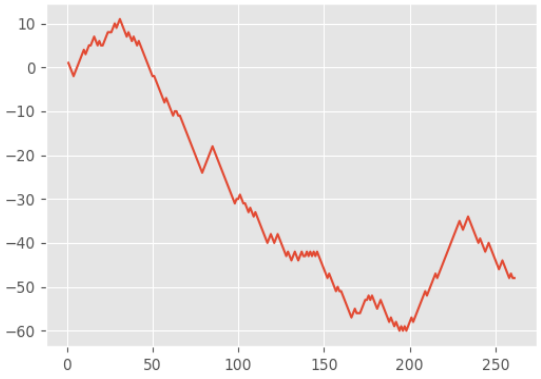


Figure 3: Graph to represent polarities of tweet

4. Discussion

In this study, we extracted tweets relating to malaria over a period of time from the social network twitter, we pre-processed the tweets extracted to eliminate unimportant tweets and redundancy, and we built a model to fit our data using machine learning approach for text classification to determine the contextual polarity of every tweet on the subject of malaria in a bid to harvest peoples opinion towards malaria and understand how well research and recent development in the aid to tackle malaria has affected the opinions of people towards the subject malaria. This study finds that though lots of ground breaking research are ongoing, awareness on malaria treatment and prevention needs to be on the increase and ground breaking research in this area needs to be communicated to the public appropriately through the appropriate authorities because tweets extracted, pre-processed and classified in this study were majorly classified as negative due to the fact that tweets tweeted were majorly about different occurrence of death, misinformation and need for donations to save a life. We hereby proposed that periodical analysis be done on the subject malaria, also expanding the source of data to closely monitor the awareness of the public and the opinions of the public on the subject malaria which would help benchmark how effective research beencarried out are affecting the public and their level of awareness on malaria prevention and treatments.

ACKNOWLEDGMENT

We acknowledge the support of Covenant University Center for Research, Innovation and Discovery, Covenant University, Ota, Nigeria.

Reference

[1] C. Ekenna, S. Fatumo, and E. Adebiyi, "In-silico evaluation of malaria drug targets," *Int. J.*, 2010.

[2] A. Golldack, B. Henke, B. Bergmann, and M. Wiechert, "Substrate-analogous inhibitors exert antimalarial action by targeting the Plasmodium lactate transporter PfFNT at nanomolar scale," *PLoS*, 2017.

[3] K. Sahan, C. Pell, and F. Smithuis, "Community engagement and the social context of targeted malaria treatment: a qualitative study in Kayin (Karen) State, Myanmar," *Malaria*, 2017.

[4] S. Hapuarachchi, S. Cobbold, and S. Shafik, "The Malaria Parasite's Lactate Transporter PfFNT Is the Target of Antiplasmodial Compounds Identified in Whole Cell Phenotypic Screens," *PLoS*, 2017.

[5] "WHO | World Malaria Report 2016," *WHO*, 2016.

[6] N. Dholakia, P. Dhandhukia, and N. Roy, "Screening of potential targets in Plasmodium falciparum using stage-specific metabolic network analysis," *Mol. Divers.*, 2015.

[7] C. Huthmacher, A. Hoppe, and S. Bulik, "Antimalarial drug targets in Plasmodium falciparum predicted by stage-specific metabolic network analysis," *BMC Syst.*, 2010.

[8] J. Teahton, "In vitro and in vivo anti-plasmodial activities of senna occidentalis roots extracts against plasmodium falciparum and plasmodium berghei," 2016.

[9] Segun Fatumo, Kitiporn Plaimas, Ezekiel Adebiyi, Rainer Konig (2011): Comparing metabolic network models based on genomic and automatically inferred enzyme information from Plasmodium and its human host to define drug targets in silico, Infection, Genetics and Evolution 11 (2011) 201–208

[10] Roll Back Malaria Partnership, "AIM 2016-2030 — Roll Back Malaria," 2016. [Online]. Available: http://www.rollbackmalaria.org/about-rbm/aim-2016-2030. [Accessed: 10-Jul-2017].

[11] D. A. Goff, R. Kullar, J. G. Newland, L. N, and M. G, "Review of Twitter for Infectious Diseases Clinicians: Useful or a Waste of Time?," *Clin. Infect. Dis.*, vol. 512, no. 10, pp. 126–9, Feb. 2015.

[12] A. J. Lazard, E. Scheinfeld, J. M. Bernhardt, G. B. Wilcox, and M. Suran, "Detecting themes of public concern: A text mining analysis of the Centers for Disease Control and Prevention's Ebola live Twitter chat," *Am. J. Infect. Control*, vol. 43, no. 10, pp. 1109–1111, Oct. 2015.

[13] O. M and Y. S, "HIV/AIDS and the Millennium Development Goals: A Public Sentiment Analysis of World AIDS Day Twitter Chat," *Int. J. HIV/AIDS Res.*, pp. 129–132, Nov. 2016.

[14] H. Kwak, C. Lee, H. Park, and S. Moon, "What is Twitter, a social network or a news media?," in *Proceedings of the 19th international conference on World wide web - WWW '10*, 2010, p. 591.

[15] R. J. P. N. N. N. Akshay Ghaisas, *International Journal of Engineering Research and Technology IJERT*, no. Vol.2-Issue 3 (March-2013). ESRSA Publ, 2012.

[16] G. G. Chowdhury, "Natural Language Processing,"

Annu. Rev. Appl. Linguist., vol. 37, no. 1, pp. 51–89, 2003.

[17] D. B. Pisoni and S. V Levi, "Some Observations on Representations and Representational Specificity in Speech Perception and Spoken Word Recognition 2 Conventional View of Speech 2.1 Background," 2005.

[18] E. Hume and K. Johnson, "A model of the interplay of speech perception and phonology," *Stud. interplay speech Percept. Phonol.*, vol. 55, pp. 1–22, 2001.

[19] M. Shapiro, "Style," pp. 203–278, 2017.

[20] K. J. Patterson, "When Is a Metaphor Not a Metaphor? An Investigation Into Lexical Characteristics of Metaphoricity Among Uncertain Cases," *Metaphor Symb.*, vol. 32, no. 2, pp. 103–117, Apr. 2017.

[21] M. Warsha, M. Choudhari, and M. Rinku Rajankar, "Introduction to Natural Language Processing With Python," 2015.

[22] A. Tamrakar and D. Dubey, "Query Optimisation using Natural Language Processing," *Int. J. Comput. Sci. Technol.*, vol. 3, no. 1, 2012.

[23] A.-H. Tan, "Text mining: The state of the art and the challenges," in *Proceedings of the PAKDD 1999 Workshop on Knowledge Disocovery from Advanced Databases*, 1999, vol. 8, pp. 65–70.

[24] S. Vijayarani, J. Ilamathi, and M. Nithya, "Preprocessing Techniques for Text Mining - An Overview," *Int. J. Comput. Sci. Commun. Networks*, vol. 5, no. 1, pp. 7–16, 2015.

[25] S. Anshika and G. Udayan, "Text Mining: A Burgeoning technology for knowledge extraction," *Int. J. Sci. Res. Eng. Technol.*, vol. 1, no. 12, pp. 022–026, 2013.

[26] A. Brahme, "A Review of Knowledge Discovery Using Text Mining and Its Applications," no. 5, pp. 291–295, 2015.

[27] Y. Zhang and P. Desouza, "Enhance the Power of Sentiment Analysis," *Int. J. Comput. Autom. Control Inf. Eng.*, vol. 8, no. 3, pp. 7–12, 2014.

[28] E. Cambria, D. Das, S. Bandyopadhyay, and A. F. Editors, *A Practical Guide to Sentiment Analysis*, vol. 5. 2017.

[29] B. Pang and L. Lee, "Opinion Mining and Sentiment Analysis," *Found. Trends Inf. Retr.*, vol. 2, no. 1–2, pp. 1–135, 2004.

[30] N. N. Yusof, A. Mohamed, and S. Abdul-Rahman, "Reviewing classification approaches in sentiment analysis," in *Communications in Computer and Information Science*, vol. 545, Springer, Singapore, 2015, pp. 43–53.

[31] R. S. Rahate and Emmanuel M, "Feature Selection for Sentiment Analysis by using SVM," *Int. J. Comput. Appl.*, vol. 84, no. 5, pp. 975–8887, 2013.

[32] U. Aggarwal and G. Aggarwal, "Sentiment Analysis : A Survey," *Int. J. Comput. Sci. Eng. Open Access Surv. Pap.*, vol. 5, no. 5, 2017.

[33] M. R. Osama, K. H. Ahmad, and A. A. Q. Dana, "Sentiment analysis as a way of web optimization," *Sci. Res. Essays*, vol. 11, no. 8, pp. 90–96, Apr. 2016.

[34] A. M. Mostafa, "An Evaluation of Sentiment Analysis and Classification Algorithms for Arabic Textual Data," *Int. J. Comput. Appl.*, vol. 158, no. 3, pp. 975–8887, 2017.

[35] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Ain Shams Eng. J.*, vol. 5, no. 4, pp. 1093–1113, Dec. 2014.

[36] K. Khan, B. Baharudin, A. Khan, and A. Ullah, "Mining opinion components from unstructured reviews: A review," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 26, no. 3, pp. 258–275, Sep. 2014.

[37] X. Fang and J. Zhan, "Sentiment analysis using product review data," *J. Big Data*, vol. 2, no. 1, p. 5, Dec. 2015.

[38] L. Zhang and B. Liu, "Sentiment Analysis and Opinion Mining," in *Encyclopedia of Machine Learning and Data Mining*, Boston, MA: Springer US, 2016, pp. 1–10.

[39] B. Shreya and P. Rupal, "A brief review of sentiment analysis methods," *Int. J. Inf. Sci. Tech.*, vol. 6, 2016.

[40] H. Zhou, J. Guo, and Y. Wang, "A feature selection approach based on term distributions.," *Springerplus*, vol. 5, p. 249, 2016.

[41] X. Ding, S. M. Street, B. Liu, S. M. Street, P. S. Yu, and S. M. Street, "A Holistic Lexicon-Based Approach to Opinion Mining," *Wsdm'08*, 2008.

[42] M. Hajjouz, "Information and Knowledge Management Opinions Mining in Facebook," vol. 6, no. 3, 2016.

[43] R. Shouval, O. Bondi, H. Mishan, A. Shimoni, R. Unger, and A. Nagler, "Application of machine

learning algorithms for clinical predictive modeling: a data-mining approach in SCT," *Bone Marrow Transplant.*, vol. 49, no. 3, pp. 332–337, Mar. 2014.

[44] Y. Baştanlar and M. Özuysal, "Introduction to machine learning," in *Methods in Molecular Biology*, vol. 1107, Humana Press, Totowa, NJ, 2014, pp. 105–128.

Author Index