

Algorithms for Intelligent Systems

Series Editors: Jagdish Chand Bansal · Kusum Deep · Atulya K. Nagar

K. G. Srinivasa

G. M. Siddesh

S. R. Manisekhar *Editors*

Statistical Modelling and Machine Learning Principles for Bioinformatics Techniques, Tools, and Applications



Springer

Algorithms for Intelligent Systems

Series Editors

Jagdish Chand Bansal, Department of Mathematics, South Asian University,
New Delhi, Delhi, India

Kusum Deep, Department of Mathematics, Indian Institute of Technology Roorkee,
Roorkee, Uttarakhand, India

Atulya K. Nagar, Department of Mathematics and Computer Science,
Liverpool Hope University, Liverpool, UK

This book series publishes research on the analysis and development of algorithms for intelligent systems with their applications to various real world problems. It covers research related to autonomous agents, multi-agent systems, behavioral modeling, reinforcement learning, game theory, mechanism design, machine learning, meta-heuristic search, optimization, planning and scheduling, artificial neural networks, evolutionary computation, swarm intelligence and other algorithms for intelligent systems.

The book series includes recent advancements, modification and applications of the artificial neural networks, evolutionary computation, swarm intelligence, artificial immune systems, fuzzy system, autonomous and multi agent systems, machine learning and other intelligent systems related areas. The material will be beneficial for the graduate students, post-graduate students as well as the researchers who want a broader view of advances in algorithms for intelligent systems. The contents will also be useful to the researchers from other fields who have no knowledge of the power of intelligent systems, e.g. the researchers in the field of bioinformatics, biochemists, mechanical and chemical engineers, economists, musicians and medical practitioners.

The series publishes monographs, edited volumes, advanced textbooks and selected proceedings.

More information about this series at <http://www.springer.com/series/16171>

K. G. Srinivasa · G. M. Siddesh ·
S. R. Manisekhar
Editors

Statistical Modelling and Machine Learning Principles for Bioinformatics Techniques, Tools, and Applications



Springer

Editors

K. G. Srinivasa
Department of Informatics, Computer
Science & Engineering
National Institute of Technical Teachers
Training and Research
Chandigarh, Chandigarh, India

S. R. Manisekhar
Department of Information Science
and Engineering
Ramaiah Institute of Technology
Bengaluru, Karnataka, India

G. M. Siddesh
Department of Information Science
and Engineering
Ramaiah Institute of Technology
Bengaluru, Karnataka, India

ISSN 2524-7565

ISSN 2524-7573 (electronic)

Algorithms for Intelligent Systems

ISBN 978-981-15-2444-8

ISBN 978-981-15-2445-5 (eBook)

<https://doi.org/10.1007/978-981-15-2445-5>

© Springer Nature Singapore Pte Ltd. 2020

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721,
Singapore

Preface

Bioinformatics is a multifaceted area involved in the development of computational methods, techniques, and software tools for biological data. For better understanding and analysing of biological data, bioinformatics integrate computer science, nature science, and mathematics fields together. It deals with the collection, modelling, manipulating of information for analysis, and visualization of biological data and in turn helps in creating novel algorithms and tools.

This edited book is used to address the topics related to bioinformatics, statistics, and machine learning. It covers the different areas of bioinformatics and helps in fulfilling the latest research gap between machine learning and bioinformatics. The main objective is to improve the knowledge in the area of bioinformatics computing and how machine learning and deep learning are involved in knowledge extraction for biological data.

This book is organized into three parts. The first part is focused on bioinformatics and machine learning, which includes six chapters. The second part highlights the protein structure prediction and gene expression analysis, which comprises four chapters. Finally, the third part illustrates the genomics and proteomics, which includes four chapters.

Bioinformatics

Chapter “[Introduction to Bioinformatics](#)” presents an overview of the bioinformatics and its importance in computational domain. This chapter also provides a brief history of bioinformatics. The author also illustrated the different computational approaches used in the biological problems.

Chapter “[Review About Bioinformatics, Databases, Sequence Alignment, Docking, and Drug Discovery](#)” presents the systematic approach to recall the terms, which are used in bioinformatics, such as databases, sequence alignment, docking, and drug discovery. This chapter also discusses the various analyses of docking and drug

discovery. Finally, the proper steps to lead the experiments for using various tools and databases with examples are addressed.

Chapter “[Machine Learning for Bioinformatics](#)” highlights various ways to analyse bioinformatics data using machine learning. The author has highlighted the different machine learning techniques in bioinformatics. They also presented two case studies using artificial neural network in bioinformatics area.

Chapter “[Impact of Machine Learning in Bioinformatics Research](#)” tells the importance of advanced tool in bioinformatics area which deals with molecular phenotypes, drug discovery, and aids in determining unfamiliar diseases. This chapter also gives a detailed overview of the impact of machine learning in the field of bioinformatics.

Chapter “[Text Mining in Bioinformatics](#)” explores the files of text mining in bioinformatics. The authors discussed the introduction to text mining and its implementation. Further, it is extended with a few case studies of text mining using different approaches.

Chapter “[Open-Source Software Tools for Bioinformatics](#)” provides a brief discussion on open-source tools used in bioinformatics area. This chapter also explains the architecture, domain, and data management issues used for solving many life science problems.

Protein Structure Prediction and Gene Expression Analysis

Chapter “[A Study on Protein Structure Prediction](#)” provides an overview of different computational methods for protein structure prediction like comparative modelling, threading, and Ab initio methods. The author has also discussed a case study on homology modelling of superoxide dismutase.

Chapter “[Computational Methods Used in Prediction of Protein Structure](#)” presents the different methods for prediction of the protein structure and their use in drug discovery. Optimization of each method is discussed here including their major significances along with the challenges.

Chapter “[Computational Methods for Inference of Gene Regulatory Networks from Gene Expression Data](#)” discusses the importance of gene regulatory networks in drug design and diagnosis of certain health conditions. Here, the author has provided a review of various approaches that are used for construction of gene regulatory networks. The details covered here pertain to properties of gene expression data and various approaches that are being used for the purpose of construction of gene regulatory networks from gene expression data.

Chapter “[Machine-Learning Algorithms for Feature Selection from Gene Expression Data](#)” details the different featured selection methods for gene data set. This chapter focuses on the different issues related to the gene expression data in the presence of noise, missing values, and its extremely high dimensionality. Finally, the author has presented a compilation of prominent machine learning methods used for feature selection from gene expression data sets.

Genomics and Proteomics

Chapter “[Unsupervised Techniques in Genomics](#)” discusses the way of handling genomics data using unsupervised techniques. This chapter also outlines a few applications of unsupervised techniques of machine learning and identifies the challenges associated with handling genomic data.

Chapter “[Supervised Techniques in Proteomics](#)” focuses on the different supervised techniques in proteomics. It dives its readers into the scope of machine learning algorithms in the study of proteins in a more extensive manner and provides examples of various real-time data sets that can be used to analyse the proteins. The author also supplements with two case studies which revolve around the application of an algorithm in a real-world data set.

Chapter “[Visualizing Codon Usage Within and Across Genomes: Concepts and Tools](#)” discusses the significance of research in biological domain. In this chapter, the author illustrated the main areas of codon usage studies with an emphasis on the tools that allow visual interpretation of the data and discussed underlying principles of different approaches, what kind of statistics lend confidence in their results, and what has to be done to further boost the field of codon usage research.

Chapter “[Single-Cell Multiomics: Dissecting Cancer](#)” provides an overview on the use of single-cell omics in cellular model of cancer and its clinical application. This chapter also highlights the potential of using multiomics approach to understand the cellular heterogeneity at multiple layers. Later, the knowledge captured from the single-cell analysis facilitated a wider understanding about disease, and in developing efficient treatments strategies are discussed.

The editors are very thankful to all the members of Springer (India) Private Limited, for the given opportunity to edit this book.

Chandigarh, India
Bengaluru, India
Bengaluru, India

Dr. K. G. Srinivasa
Dr. G. M. Siddesh
S. R. Manisekhar

Contents

Bioinformatics

Introduction to Bioinformatics	3
S. R. Manisekhar, G. M. Siddesh and Sunilkumar S. Manvi	
Review About Bioinformatics, Databases, Sequence Alignment, Docking, and Drug Discovery	11
P. Lakshmi and D. Ramyachitra	
Machine Learning for Bioinformatics	25
K. Aditya Shastry and H. A. Sanjay	
Impact of Machine Learning in Bioinformatics Research	41
E. Naresh, B. P. Vijaya Kumar, Ayesha and Sahana P. Shankar	
Text Mining in Bioinformatics	63
Minal Moharir and Preetham Maiya	
Open-Source Software Tools for Bioinformatics	75
T. Gururaj and A. P. Pavithra	
Protein Structure Prediction and Gene Expression Analysis	
A Study on Protein Structure Prediction	95
Biboshan Banerjee, G. M. Siddesh and K. G. Srinivasa	
Computational Methods Used in Prediction of Protein Structure	119
Poulami Majumder	
Computational Methods for Inference of Gene Regulatory Networks from Gene Expression Data	135
Nimrita Koul and Sunilkumar S. Manvi	
Machine-Learning Algorithms for Feature Selection from Gene Expression Data	151
Nimrita Koul and Sunilkumar S. Manvi	

Genomics and Proteomics

Unsupervised Techniques in Genomics	165
Mrinmoyee Bhattacharya	
Supervised Techniques in Proteomics	189
Vasireddy Prabha Kiranmai, G. M. Siddesh and S. R. Manisekhar	
Visualizing Codon Usage Within and Across Genomes: Concepts and Tools	213
Bohdan Ostash and Maria Anisimova	
Single-Cell Multiomics: Dissecting Cancer	289
Janani Sambath, Krishna Patel, Sewanti Limaye and Prashant Kumar	

About the Editors

K. G. Srinivasa is currently working as a Professor in the Department of Informatics, Computer Science & Engineering of National Institute of Technical Teachers Training and Research, Chandigarh. He was awarded a Ph.D. in Computer Science and Engineering from Bangalore University in 2007. He has received various awards, including the All India Council for Technical Education – Career Award for Young Teachers; Indian Society of Technical Education – ISGITS National Award for Best Research Work Done by Young Teachers; Institution of Engineers (India) – IEI Young Engineer Award in Computer Engineering; the ISTE's Rajarambapu Patil National Award for Promising Engineering Teachers in 2012; and a Visiting Scientist Fellowship Award from IMS Singapore. He has published more than 100 research papers in international journals and conferences, and has authored three textbooks: *File Structures using C++*, *Soft Computing for Data Mining Applications* and *Guide to High Performance Computing*. He has also edited research books in the area of cyber-physical systems and energy-aware computing. He has been awarded a BOYSCAST Fellowship by the DST to conduct collaborative research with the Clouds Laboratory at the University of Melbourne. He is the Principal Investigator for several AICTE, UGC, DRDO, and DST funded projects. He is a senior member of IEEE and ACM. His research areas include data mining, machine learning, and cloud computing.

G. M. Siddesh is currently working as an Associate Professor at the Department of Information Science & Engineering, Ramaiah Institute of Technology, Bangalore. He has authored books titled: *Network Data Analytics: A Hands-On Approach for Application Development*, *Statistical Programming in R and Internet of Things*. He has edited books titled: *Cyber Physical Systems - A Computational Perspective*, *Emerging Research Surrounding Power Consumption and Performance Issues in Utility Computing and The Rise of Fog Computing in the Digital Era*. He has published numerous research papers in reputed international journals and conferences. His research interests include data analytics, Bioinformatics, distributed computing, grid/cloud computing, and IoT.

S. R. Manisekhar received his M.Tech. degree from Bharathidasan University, Tiruchirappalli, and B.E. degree from Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal. He is currently an Assistant Professor at the Department of Information Science & Engineering, Ramaiah Institute of Technology, Bangalore. He is a member of ISTE. He has published a numerous research papers and book chapters. His research interests include data science, data analytics, and software engineering & bioinformatics.

Bioinformatics

Introduction to Bioinformatics



S. R. Manisekhar, G. M. Siddesh and Sunilkumar S. Manvi

1 Introduction

The human species, being a self-aware and intelligent life form, has always sought to find how the human body works, how our physical structure came to be, what are the chemical processes that came about along with complex molecular interactions and physiological mechanisms that were discovered and the development and evolution of the body.

Many comparisons were made between the human body and the intricacy of man-made machines and tools like timepieces, mechanical toys and pneumatic machines, even stretching to the extent of mills, factories and assembly lines during the Industrial Revolution.

Soon, this interest began to wander towards other species, their interactions with nature, what was their physiology and how they co-inhabited, interacted and adapted to the same environment as us. By studying countless species, we were able to gather some information about all of this, but it was becoming more and more difficult. As more species were discovered, the data size became larger and more complex, making it difficult to analyze.

With the dawn of the computer age, computers were becoming more accessible beyond the electrical engineering departments. Many biologists realized that they could use various computational methods to solve the data-intensive biological problems that came up. Soon, ecologists, biotechnologists, agricultural scientists and medical researchers could identify and successfully classify every gene, every

S. R. Manisekhar (✉) · S. S. Manvi
School of C&IT, REVA University, Bangalore, India

G. M. Siddesh
Department of Information Science & Engineering, Ramaiah Institute of Technology,
Bangalore, India

protein, every control signal—and not only for humans. Bioinformatics became the guiding path to gain this knowledge.

2 Bioinformatics: What Is It?

The National Centre for Biotechnology Information (NCBI) defines Bioinformatics as the interdisciplinary field of science in which biology, information technology and computer science merge together, to allow new biological discoveries and insights and also help create a global perspective from which the principles of Biology can be identified. The Bioinformatics field [1] is heavily dependent on the field of artificial intelligence (AI) to handle the complexity and reduce human error in any processes involved, which does not require human intervention.

The major objectives of this field are:

- Handle the large datasets with humongous data size produced by all biological studies.
- Develop and test tools to handle this huge amount of generated data and the complimentary increase in complexity.
- In silico and wet lab results interpretation.

The large data size presents both: an opportunity and a challenge for data mining [2]. The traditional computer algorithms are now failing to identify and address many of the highly challenging and fascinating problems of sequence analysis [3, 4]. This is due to the intricate and complex nature of the biological systems.

The main fundamental ideology to learn the theory automatically from the data by a process of inference is enforced and aided by the machine learning perspective (e.g. neural networks, K-means clustering, vector support machines, Markov chains and statistical models), ideal for the case of humongous data sets. Thus, it is another viable method to the traditional and conventional methods.

So, to summarize, searching biological databases, juxtaposition of sequences, and answering complex biological and bio-medical queries come under the huge umbrella, that is, Bioinformatics.

3 Importance of Bioinformatics

Bioinformatics is used to comprehend and analyze the regulation of cells, the function of genes, disease and drug design. Without the analysis methods provided by Bioinformatics, the study of all the massive amount of system-generated data could not have resulted in any meaningful interpretation. The modern developments in the field of biology and medicine could not have been possible without Bioinformatics. The demand for this skill is higher than ever, as more and more advancements are made in the management and analysis of this data.

Bioinformatics is mainly required to generate and develop software tools to understand basic biological data and make some useful interpretations. Bioinformatics has become a lifeline for many biologists and also in various fields of biology. Techniques such as image processing and signal processing are used to extract meaningful data from a given raw data, in the field of molecular biology. It also is used to mine the literature of the biological field and query any new data generated. Bioinformatics is very useful for the expression and regulation of gene and proteins. It helps us to understand the basic fundamentals of evolution in molecular biology. It helps study the bio-pathways and networks that are important to systems biology. It aids in the modelling of RNA, DNA, protein structures and interactions of molecules. Research is done in proteomics (study of proteomes), genomics (study of genes) and systems biology.

If data is presented, but it is in the raw form, then it is unusable, even for the professional researchers. Bioinformatics makes it very easy to understand the biological mechanisms of an organism [5]. Due to the advancements in Bioinformatics, clinical medicine and biological research has greatly improved. For example, scientists can find cures against diseases such as cystic fibrosis which are hereditary or diseases like cancer or heart disease which are acquired. It has become possible to identify or analyze the molecular constitution and basis of a disease. This has allowed pharmaceutical companies to design drugs which specifically target only the affected genes, thus minimizing any precedence side effects that non-specific generic medicine would have had. This study of developing effective clinical medicine has become a highly specialized field in itself called **Pharmacogenomics**, whose backbone is, again, Bioinformatics. The knowledge of the genetic mechanisms of a disease allows doctors to take diagnostics tests more accurately, allowing the possibility to use genes for curing diseases instead of medicines, in the near future.

4 Emergence of Bioinformatics

The term “Bioinformatics” was coined in 1970 by Hesper and Hogeweg to refer to biotic system-related information processes, which drew parallelism between biochemistry and Bioinformatics as a field.

The concept of Bioinformatics seems to have emerged recently to assist in compilation and study of biological sequences; however, Bioinformatics was conceptualized over 50 years ago, when DNA sequencing was not possible. Computational methods laid the foundation for protein sequence analysis, in the early 1960s (notably, biological sequence database and substitution models, and also de novo sequence assembly). DNA analysis emerged due to (1) molecular biological methods and (2) computer software and technology advancements which saw the increase in computational power due to miniaturization of parts, as well as custom software built to handle bioinformatic calculations and tasks. It is widely accepted that the mathematization of biology on many fronts led to the development of this new field of Bioinformatics. The machines only helped accelerate the entire process.

The huge accumulation of metabolomics, genomics and proteomics data, as well as their need for analysis, storage organization, annotation, systemization, and integration into bio-databases, were the main reason to create this field.

Between the 1990s and 2000s, there was an exponential increase in the raw data produced. This phenomenon happened due to improvement in technology used for sequencing as well as the reduced costs. “Big Data”, as it was called, created new challenges for the mining and management of data that needed to be solved. This required the expertise of computer scientists, and thus, computer science merged into the ever-inclusive field of Bioinformatics. Big Data had and is having a profound effect on the reproducibility and predictability of Bioinformatics results. Thus, the new generation of bioinformaticians is seeing the inclusion of related computer science topics in their curriculum.

5 Computational Biology and Bioinformatics: A Comparison

To understand the basic difference between these fields, let us look at an overview of the basic definition of these two connotations. **Bioinformatics** involves the application research and development of computational tools and aims to organize visualize, archive store and acquire data (not in this order). The data may be biological, behavioural, medical or health. **Computational Biology** involves the development and application of analyzing methods for data, mathematical modelling and simulation techniques to study social, behavioural and biological systems. Table 1 discusses some of the key difference [6].

Let us take a, suppose we have to execute\test an engineering method or model, we would design it such that it has certain performance characteristics, and performs as intended. We validate this method and test it solve a class of similar problems with

Table 1 Key difference between Bioinformatics and computational biology

Bioinformatics	Computational biology
Only computational, but involves the acquiring and storing of data	Only computational
Data analysis of biological data. Computer is used to gather, store integrate information which is then used for gene-based drug development and discovery	Mechanistic understanding of part or whole of a biological process. New algorithms are developed with which large biological information can be analyzed
Provides an answer by statistical inferences	Provides an answer by modelling processes via ordinary or partial differential equations
Can be called a subfield of computational biology. Equal focus on biological aspect and the engineering of data	More of a general umbrella field which includes many subfields. Main focus is gathering more information and biological knowledge

the help of software (that is, testing the model or method with a large amount of data to prove correctness and validate it) and we then proceed to write papers about this method, then this would be an example of Bioinformatics.

When a method is used to answer a biological question, the success of this method does not depend on the computational methods\tools applied. It is about whether the new discovery or the answer is true, and is validated or not, and if the evidence is up to the standard of evidence expected by the biological community. This would be a case of computational biology.

However, there is not a fine difference as of today, between these fields. With the increase in data and the technology advancements, the line drawn between them is becoming a bit blurred. The major differences are still present, but the similarities between these fields are increasing. Thus, one may argue that the difference between these fields may vary for every researcher.

6 Computational Approaches to Biological Problems

One thing that drastically changes as there is increase in raw data is the way it is analyzed and studied. Greater computational power is required, and new computational tools and methods have to be developed to ensure greater accuracy in prediction of outcomes. Especially in the field of biology, where getting a protein prediction right can save thousands of lives. Thus, computational methods are an integral part of the field as a whole, used for the management and study of data, and also how it is represented. It is interwoven into the fabric of biology.

Let us take a look at some of the subfields of computational biology, and what are the advancements made in those fields. We will also look at how they approach the biological problems.

- **Computational Bio-modelling:** It is a field which tackles biological problems by building computer models. It uses simulations to asses and evaluates the complexity of bio-systems. The method to approach such a problem requires the use of highly specialized algorithms and visual software [7]. This tests if the system is robust, by observing the changes in the system when exposed to different environments. Computational bio-modelling allows multiple users to study the huge amount of data generated.
- **Computational Genomics:** This field involves the study of genomes of cells and organisms [8, 9]. One example is the human genome project, which aims to acquire data about the entire human genome, it will allow the doctors to analyze the genome of a patient and this further opens up the doorway for creating target-specific and personalized medicine, based on the genetic pattern of the patient. Sequence homology is used to compare genomes, by comparing and studying various nucleotide structures that are acquired from the same ancestor. This field is still in development and researchers are trying to understand the non-coding

areas of the genome via large consortia projects like the Roadmap Epigenetics Project and various other computational and statistical methods.

- **Computational Neuroscience:** This field studies the properties and structure of the nervous system and can create tools for practical application by analyzing brain data [10]. Models of the brain are used to analyze the different aspects of the nervous system. Some of the brain models include **Realistic brain models** which represent the brain in detail, down to the cellular level. These models, even though so intricate and detailed, are very prone to errors; as it does not include the cellular structures, the scientists do not know about. These are very costly to implement and require a huge amount of computational power. The other is the **Simplifying Brain Model**, which assesses physical properties of neurological systems. It also reduces the potential error by a higher factor than the realistic brain model.
- **Computational Pharmacology:** The book “Computational Biologists: The Next Pharma Scientists?” Written by Price, Michael, defines computational pharmacology as “*the study of the effects of genomic data to find links between specific genotypes and diseases and then screening drug data*” [11]. The pharma companies required a shift from the traditional use of Excel worksheets to compare data related to the effectiveness of drugs, which led to the development of this sub-field. Computational methods such as machine learning are used to analyze this huge collection of data, leading to a more meaningful and efficient comparison of notable points leading to better drug development.
- **Computational Evolutionary Biology:** This field makes the use of DNA data to reconstruct tree of life. It uses a computational method known as **Computational phylogenetics** (use of algorithms and methods for phylogenetic analysis). It also fits population models to data about DNA and evaluates demographic and selective history. The prediction of which evolutionary system is likely to evolve first is made by building population genetics from first principles.
- **Cancer Computational Biology:** This field aims to determine the future mutations in the cancer cells [12]. It does this by applying an algorithmic approach to analyzing data, leading to the use of high throughput measurement. Data is collected from the RNA, DNA and other structures. It helps in determining the causation of tumours and cancer, and how is it connected to the human genome.
- **Computational Neuropsychiatry:** It is an emerging field that uses computers to model the brain, to detect the mechanisms involved in mental disorders. It also helps to understand how mental functions occur and dysfunctions are caused by understanding the neuronal circuits [13].
- **Computational Anatomy:** This field involves the imaging of biological and anatomical structures, via technologies such as MRI, to obtain dense 3D measurements [14]. It also requires the implementation of mathematical and data analytical modelling methods for simulation of structures.

7 Conclusion

Bioinformatics domain helps in drawn of inference for a biological database using computational techniques. This chapter provides a brief introduction to the Bioinformatics and its importance in living organisms. Later, the author has illustrated the key differences between computational biology and Bioinformatics. The different computational approaches show how Bioinformatics deals with different datasets for resolving issues of different biological problems.

References

1. Can T (2014) Introduction to bioinformatics. *Methods Mol Biol* 1107:51–71
2. Manyika J, Chui M, Brown B et al (2014) Big data: the next frontier for innovation, competition, and productivity 2011
3. Chen XW, Gao JX (2016) Big data bioinformatics. *Methods* 111:1–2
4. Greene CS, Tan J, Ung M, Moore JH, Cheng C (2014) Big data bioinformatics. *J Cell Physiol* 229(12):1896–1900
5. Greene CS, Troyanskaya OG (2011) PILGRM: an interactive data-driven discovery platform for expert biologists. *Nucleic Acids Res* 39(Web Server issue):W368–W374
6. Nair AS (2007) Computational biology & bioinformatics: a gentle overview. *Commun Comput Soc India* 2
7. Kitano H (2002) Computational systems biology. *Nature* 420(6912):206–210
8. Koonin EV (2001) Computational genomics. *Curr Biol* 11(5):R155–R158
9. Cristianini N, Hahn M (2006) Introduction to computational genomics. Cambridge University Press
10. Trappenberg TP (2002) Fundamentals of computational neuroscience. Oxford University Press Inc, Oxford, p 1
11. Price M (2012) Computational biologists: the next pharma scientists. *Science Careers*
12. Yakhini Z, Jurisica I (2011) Cancer computational biology. *BMC Bioinf* 12:120
13. Dauvermann MR, Whalley HC, Schmidt A, Lee GL, Romaniuk L, Roberts N, Johnstone EC, Lawrie Sm, Moorhead TW (2014) Computational neuropsychiatry—schizophrenia as a cognitive brain network disorder. *Front Psychiatry* 5:30
14. Grenander U, Miller MI (1998) Computational anatomy: an emerging discipline. *Q Appl Math* 56(4):617–694

Review About Bioinformatics, Databases, Sequence Alignment, Docking, and Drug Discovery



P. Lakshmi and D. Ramyachitra

1 Bioinformatics and Databases

Biological data was to be interpreted for the purpose of analysis and processes the data in various forms that to be stored and retrieved from databases. The importance of a particular gene or protein recognized and focused on genetics. The particular or common domain which helps to access the wide variety of resources has to be authenticated. Bioinformatics uses to launch the applications of computer science and information technology into the field of molecular biology. The raw data can be stored in the database to organize and manipulate the data for extraction that can be easily utilized by the researchers.

Figure 1 shows some layers of the bioinformatic processes. The detail of the sequence information depends upon the particular organisms that are stored in the biological databases. It helps to understand the new sequence genome and protein sequence for the purpose of comparison and identification between them become easier. For that, various tools are to be addressed.

The knowledge of genomic and bioinformatics tools is very important for the scientist in veterinary and animal science. By using these tools, the improvement of productivity of farm animals will be possible in the future. In the sequence cost reduction, all individual's population tested that might be genotyped and sequenced.

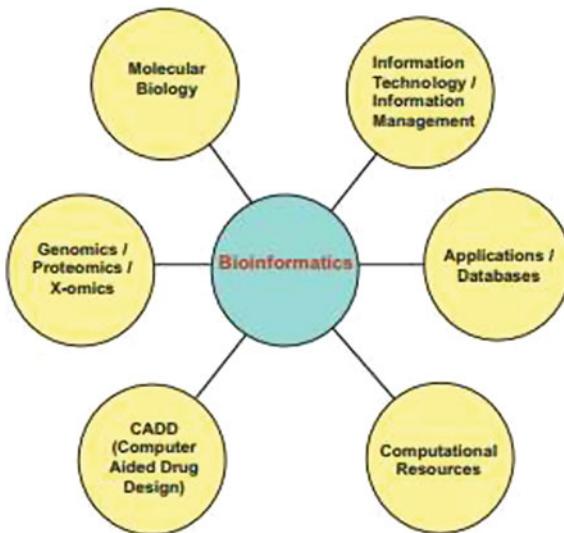
1.1 Bioinformatics in Animal Science

The knowledge of bioinformatics helps to understand the storing methods, biological data analysis such as sequence of protein, nucleic acid, function, structure, pathways,

P. Lakshmi · D. Ramyachitra (✉)

Department of Computer Science, Bharathiar University, Coimbatore, Tamilnadu, India

Fig. 1 Overview of bioinformatics



and interaction of genetics [1]. Algorithms, databases, statistics, control and system theory, circuit theory, structural biology, information and computation theory, Web technologies, soft computing, software engineering, data mining, image processing, artificial intelligence are included in bioinformatics, which help to improve the development of the living organisms.

1.2 *Genome Sequencing*

The requirement of homology modeling is the part of the sequence alignment and to predict the structure of a protein to know the homology structure. This alignment helps to predict the difference between the structure and template of the sequence. BLAST and FASTA are the basic operations for sequence alignment. The steps of alignment sequences are the identification of sequence, search in database, homolog detection, sequence alignment, add structural information, perform required operation. X-ray crystallography, NMR, and reason version of experiment instruments are discussed to know the information of isolation. Some of the computational algorithms also addressed to align the sequences in an effective manner. Three types of alignments available that is single, pairwise, and multiple sequences.

While DNA sequencing and alignment, measurement of gene expression techniques, modeling studies for protein folding patterns and datasets requires large storages like terabytes, modern computational and machine learning capabilities helps to reduce the complexity for the process of analysis, storage and interpretation [2]. DNA sequencing is critical for any genomic experiment. It is costly, demands a lot of time, and could be still technically difficult. New emerging technologies were

successful in solving most of these problems although for animal genomic studies' cost is still a limitation. Any information generated downstream of the sequencing will be determined by it; for this, the sequencing process should be as much accurate as possible [3]. Since then, the genomes from different species including livestock animals have been sequenced. A lot of further investigation should be performed to understand more about genes, its functionality, and disease involvements [4].

1.3 *Genome Assembly*

Genome assembly consists in ordering a bulky number of DNA fragments to reconstruct the original DNA structure from which they were originated. Although reading a book by one letter at a time is possible, reading the whole genome by one nucleotide at a time is still unfeasible. DNA sequencing technologies only could read small DNA fragments between 20–1000 nucleotides [5]. Presently, biologists can identify short reads with unknown genomic positions. For this reason, genome assembly reminds a puzzle with billion of pieces, an ambitious mathematical and computational problem. The main idea is to produce a large numbers of reads from many copies of the same genome, which represents a huge overlapping puzzle [6]. The complex problem can be simplified in two parts: reads generation and fragments assembly, which present a biological and algorithm problem, respectively [7].

2 Databases

Information on protein and gene sequences is stored in the databases. They can be easily manipulated, and the databases are located in different places. Day-to-day, the information is exchanged, updated, and synchronized [8]. According to the gene, the database search depends on the following information which is given below:

- Genes homologous are the distribution of taxonomic, frequencies of allele, and synteny.
- Information of genome is location of chromosome, introns, regions regulatory, and domain sharing.
- Information belongs to the structures which are protein structure connection, types of fold, domain structures.
- Information of expressions is the particular tissues of a specific expression, stage developments, phenotypes, and diseases.
- Information of functions is the molecular and enzymatic function, role of pathway or cellular, localization, and disease roles.

2.1 Database of Gene and Nucleic Acid

- BioGPS—Database of Genome Sequence and the GenBank division
- HGMD—Database with the collection of human mutated gene (human gene mutation database)
- COSMIC—Information about the mutated somatic details in cancer (catalogue of somatic mutations in cancer)
- DBEST—Division of GenBank—Collection of gene information storage
- HGVS—Group of genome cluster society with variety of different human genome differentiability (Human Genome Variation Society)
- DDBJ DNA—Data Bank of Japan, Data storage of DNA collection located in Japan
- IMGT/GENE-DB—the international immune genetics information system for immunoglobulin
- ENSEMBL—Projects among EMBL and EBI, Sanger Institute, of selected eukaryotic genomic annotation
- EMBL (European Molecular Biology Laboratory)—laboratory of molecular biology located in Europe
- ENA—European Nucleotide Archive
- INSDC—International Nucleotide Sequence Database Collaboration
- GenAtlas—Database of human genes
- GenBank—Collection of gene information storage in the USA
- GHR—Genetics Home Reference
- GSDB—Database of Genome Sequence
- OMIA—Online Mendelian Inheritance in Animals
- OMIM—Online Mendelian Inheritance in Man
- TGI—the Gene Index Project.

2.2 RNA Specific Resources

- miRBase—microRNA database
- ncRNA Expression Database (NRED)—Database of expression data on human and mouse long ncRNAs
- piRNABank—Web resource on classified and clustered piRNAs
- Rfam—RNA families' database of alignments and CMs
- Sno/scaRNAbase—Curated repository for Small Nucleolar RNAs and small Cajal body-specific RNA
- RNACentral—Resource to organize data for non-protein coding RNA genes
- TarBase—Curated database of experimentally supported microRNA targets.

2.3 Protein Databases

- BRENDA—the comprehensive enzyme information system
- APID—Agile Protein Interaction DataAnalyzer
- BIND—Binding Database
- BioGRID—Biological General Repository for Interaction Datasets
- CATH—Protein structure classification
- ExPASy—Proteomics server
- ExplorENZ—the enzyme database
- HPRD—Human Protein Reference Database
- Human Proteinpedia—Community portal for sharing and integration
- HUPO—Human Proteome Organization
- iHOP—Information Hyperlinked over Proteins
- IUPHAR—International Committee of Pharmacology Committee on Receptor
- MEROPS—Peptidase database
- MINT—Molecular INTeraction database
- NCBI—Protein database neuropeptides
- PDB—Protein Data Bank
- PhosphoSitePlus—A Protein Modification Resource
- PIR—Protein Information Resource
- PMDB—Protein Model Database
- SCOP—Structural Classification of Proteins
- STRING—Search Tool for the Retrieval of Interacting Genes/Proteins
- SWISS-PROT ENZYME TRANSFAC—Transcription Factor Database
- TrEMBL—Computer-annotated supplement to Swiss-Prot
- UniProt—the Universal Protein Resource
- DIP—Database of interacting proteins.

2.4 Databases on Mutations and Variations

- dbSNP—<http://www.ncbi.nlm.nih.gov/SNP>
- dbVar—Database of Genomic Structural Variation
- ENCODE—ENCyclopedia Of DNA Elements
- HapMap—international HapMap Project
- HGBASE—Human Genic Bi-Allelic SEquences <http://hgbase.cgr.ki.se>
- Human Genome Segmental Duplication Database and Human Structural Variation Database
- The SNP Consortium (TSC)—<http://snp.cshl.org>

2.5 Bioinformatics Analysis Tools

- 2ZIP—leucine zipper prediction server
- GenomeNet—Bioinformatics tools
- EBI—Toolbox
- RepeatMasker—screens DNA sequences for interspersed repeats and low complexity DNA sequences
- SOSUI—transmembrane prediction server
- SMART—Simple modular architecture research tool
- Splign—utility for converting cDNA to genomic or spliced sequence align.

In computational biology, sequence analysis is the most primitive operation. This operation performs the similarity of the biological sequences, variation of the medical analysis, and genome mapping processes. Sequence analysis indicates subjecting a DNA or peptide sequence to the alignment of sequence, databases of sequence, repeated search of sequences, and bioinformatic methods on a computer.

3 Docking and Drug Discovery

Docking uses to confirm ligand binding to receptor, and usually receptor is bigger than the molecules. This information includes the coordination of the ligand atoms and finds the lowest energy bind site of the docking configuration. FLEX and auto-docking example programs will be shown in a later chapter of this paper. The aim is to predict the conformation bound and the affinity binding. The interaction between two molecules is determined, and the overall minimum energy of complex formation was to be found with the best orientation of ligand binding. Various bioinformatic tools are helpful for disease management, diagnosis, and drug discovery. Sequencing is enabled to identify the disease and drug discovery by scientists. Mutation and drug are all identified and experimented by utilizing different computational tools. Drug targets decide the suitable drug entry into the pipeline of drug development with the help of bioinformatic tools.

3.1 BLAST

In the database, sequence similarity can be identified between a query sequence and sequences by using the basic local alignment search tool (BLAST) operation. To allow the putative gene identification, sequence homology was to be detected, which helps to determine the protein or gene relations between them. BLAST operation uses to identify the performance of similarity, alignment, homology, annotation between the sequences of genes or proteins. The figure shows some of the examples of BLAST operation [9].

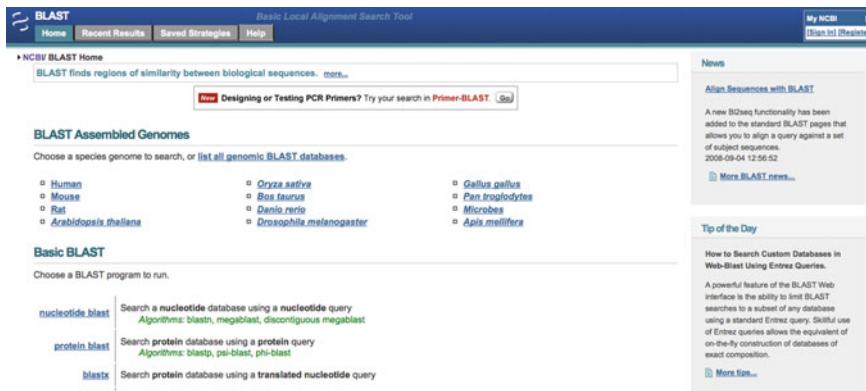


Fig. 2 Web interface of the BLAST at NCBI.100

Figure 2 shows the next-generation first operation of the sequence, i.e., Web interface of the BLAST operation which is performed in the NCBI database contains some BLAST operation with sequence.

Tab	Explanation
Home	Home page of BLAST link
Recent results	Results of the BLAST link, and we can retrieve the current browser session
Saved strategies	In my NCBI account, input parameters of BLAST have been saved
Help	Documentation of BLAST help will be listed

In Figure 3, default parameters are given to perform the BLAST operation. Steps of BLAST operation are:

1. Download the sequence and save it.
2. In NCBI Web server, click the “nucleotide BLAST”.
3. Browse the sequence file for open or copy and paste the sequence from the source.
4. Select Job title—BLAST.
5. From “Choose Search Set” tab, database modification as “Reference mRNA sequences (refseq_rna)”. Similar sequences are retrieved from selected programs.
6. Go to the check box with “show results in a new window” and click “BLAST”.

Figure 4 shows the performance of the BLAST operation with unknown sequence of relevant RNA database search. In Fig. 5, the result of the BLAST operation will be shown after verifying the ID and all with respect to the relevant input values.

Basic BLAST

Choose a BLAST program to run.

<u>nucleotide blast</u>	Search a nucleotide database using a nucleotide query <i>Algorithms: blastn, megablast, discontiguous megablast</i>
<u>protein blast</u>	Search protein database using a protein query <i>Algorithms: blastp, psi-blast, phi-blast</i>
<u>blastx</u>	Search protein database using a translated nucleotide query
<u>tblastn</u>	Search translated nucleotide database using a protein query
<u>tblastx</u>	Search translated nucleotide database using a translated nucleotide query

Fig. 3 NCBI with different BLAST programs

3.2 FASTA

It is a representation of character that may either nucleotide or amino acid sequences using letter codes individually. Process of sequences and its names with comments were to be proceeded known as FASTA, which originates with its software package [10].

The following steps were to be performed for FASTA operation:

1. The sequence files were to be added or uploaded from the databases with the file extension format is FASTA (e.g., sequence FASTA).
2. All sequence files should be unique, and no other information stored in FASTA file sequences.
3. If it is not aligned properly, then the orientation will be affected the FASTA result.

The process of designing a medication with the target molecule is known as drug designing. The small molecule is a ligand that inhibits or activates the biological target molecule's output in the therapeutic effect [11]. However, a single regulatory approach is a difficult task in marketing authorization application (MAA) in all country. Figure 6 represents the levels of drug discovery with regulations.

Figure 7 shows the drug designing method and its types, which are given below:

Traditional Drug Designing

- Trial and error method
- Preliminary methods of drug designing
- Checking the efficacy of the chemicals by giving directly to the animals
- Forward pharmacology.

The screenshot shows the NCBI BLAST search interface. At the top, there's a navigation bar with links for Home, Recent Results, Saved Strategies, and Help. Below this is a sub-navigation bar for the NCBI/BLAST/blastn suite, with tabs for blastn, blastp, blastx, tblastn, and tblastx. A message states: "BLASTN programs search nucleotide databases using a nucleotide query." Below this is a section titled "Enter Query Sequence" with a text input field for "Enter accession number, gi, or FASTA sequence". To the right of the input field are "Clear" and "Query subrange" buttons, along with "From" and "To" fields for specifying a range. Below the input field, there's a link to "more...". Further down, there's a "Or, upload file" section with a "Browse..." button and a "Job Title" field containing "blastn search D. yakuba / Refseq RNA". There's also a field for "Enter a descriptive title for your BLAST search". A checkbox for "Align two or more sequences" is present. The next section, "Choose Search Set", includes a "Database" dropdown set to "Reference mRNA sequences (refseq_rna)" (highlighted in yellow), "Organism Optional" dropdown, and an "Exclude" checkbox. The "Entrez Query Optional" section contains an input field for "Enter an Entrez query to limit search". The final section, "Program Selection", allows "Optimize for" options: "Highly similar sequences (megablast)", "More dissimilar sequences (discontiguous megablast)", and "Somewhat similar sequences (blastn)" (selected). A link "Choose a BLAST algorithm" is also provided.

Fig. 4 BLAST search for unknown sequence with NCBI RefSeq RNA database

Rational Drug Designing

- Ligand-based drug designing
- Indirect drug designing
- Knowledge of small molecules binding to its targets
- Pharmacophore-based models
- QSAR.

The screenshot shows the NCBI BLAST search results page. At the top, there's a navigation bar with links for Home, Recent Results, Saved Strategies, and Help. Below the navigation bar, the text 'Basic Local Alignment Search Tool' is displayed. The main content area shows a search job titled 'Job Title: blastn search D.yakuba / Refseq RNA search'. Underneath the title, the status is listed as 'WAITING'. Detailed information about the request includes:

- Request ID: 56RFPSX1012
- Status: Searching
- Submitted at: Tue May 22 17:17:42 2007
- Current time: Tue May 22 17:17:45 2007
- Time since submission: 00:00:03

At the bottom of the page, a message states: "This page will be automatically updated in 13 seconds until search is done". A footer bar at the very bottom contains links for Copyright, Disclaimer, Privacy, Accessibility, Contact, and Send feedback on new interface.

Fig. 5 BLAST search for known sequence with NCBI RefSeq RNA database

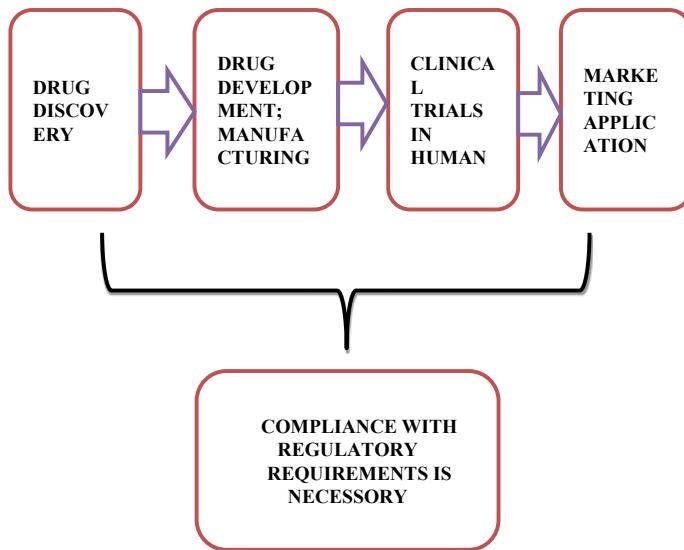


Fig. 6 Diagrammatic representation of basic regulation

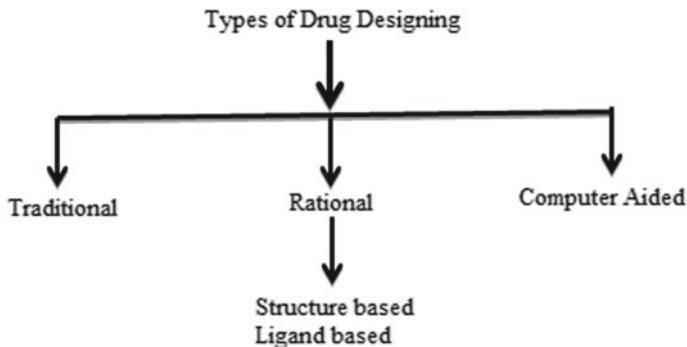


Fig. 7 Types of drug design

Structure-Based Drug Designing

- Direct drug designing
- Knowledge on biological targets and its 3D properties from X-ray crystallography models contains three methods, that have been followed
 1. High-throughput virtual screening
 2. de novo designing of small molecules (novel)
 3. Optimization of existing ligands.

Computer-Aided Drug (CAD) Design

- A dedicated regulation of computational method is CAD—computer-aided drug design. It stimulates the interaction of drug receptor [12].
- Molecular mechanics or molecular dynamics are essential to predict the conformation of the small molecule and the target protein.
- Bioinformatics tools and databases have closely related to CAD design methods.

3.3 Drug Bank

It has unique bioinformatics and cheminformatics that have pharmacological and pharmaceutical data along with their drug targets. Each drug contains more than 200 data fields. The database contains totally 8261 drugs in it.

3.4 Drug Approval Process in India

If the company wants to import or manufacture a new drug, then it has to get permission and license authority from DCGI. With the act of schedule Y of drugs

and cosmetics 1940 and rules 1945, the document has to be submitted. Based on the guidelines of schedule Y, clinical trials were engaged to prove which depend on public health, permission granted to import new drugs to other countries, after authorization of that drug properly [13]. Figures 8 and 9 are the web page of the drug bank. Here we can search the details of the drug with its relevant ID and its required information.

The screenshot shows the homepage of DrugBank Version 5.0. At the top, there's a banner with the text "Get DrugBank to go! The DrugBank app for iOS and Android is coming soon." and a "Sign up to get early access" button. Below the banner, the DrugBank logo is prominently displayed next to the text "DrugBank Version 5.0". A detailed description follows: "The DrugBank database is a unique bioinformatics and cheminformatics resource that combines detailed drug (i.e. chemical, pharmacological and pharmaceutical) data with comprehensive drug target (i.e. sequence, structure, and pathway) information. The database contains 8261 drug entries including 2021 FDA-approved small molecule drugs, 233 FDA-approved biotech (protein/peptide) drugs, 94 nutraceuticals and over 6000 experimental drugs. Additionally, 4338 non-redundant protein (i.e. drug target/enzyme/transporter/carrier) sequences are linked to these drug entries. Each DrugCard entry contains more than 200 data fields with half of the information being devoted to drug/chemical data and the other half devoted to drug target or protein data." To the right, there's a link "More about DrugBank" with a blue icon. Below this, there's a search bar with the placeholder "Search DrugBank" and a dropdown menu set to "Drugs", followed by a search button with a magnifying glass icon. A note at the bottom left states: "DrugBank is offered to the public as a freely available resource. Use and re-distribution of the data, in whole or in part, for commercial purposes (including internal use) requires a license. We ask that users who download significant portions of the database cite the DrugBank paper in any resulting publications." It also lists the citation: "Law V, Knox C, Djoumbou Y, Jewison T, Guo AC, Liu Y, Maciejewski A, Arndt D, Wilson M, Neveu V, Tang A, Gabriel G, Ly C, Adamjee S, Dame ZT, Han B, Zhou Y, Wishart DS. DrugBank 4.0: shedding new light on drug metabolism. Nucleic Acids Res. 2014 Jan 1;42(1):D1091-7. 24203711." A "View full article online" link is also present.

Fig. 8 Application of drug bank version 5.0

The screenshot shows a detailed view of a DrugCard for Genistein (ID DB01645). The top navigation bar includes links for "Browse", "Search", "Downloads", "About", "Help", and "Contact Us". On the right, there are "Search" and "Drugs" buttons. The main content area is titled "Identification". It displays the following details:

- Name: Genistein
- Accession Number: DB01645 (EXPT01582)
- Type: Small Molecule
- Groups: Investigational
- Description: An isoflavonoid derived from soy products. It inhibits protein-tyrosine kinase and topoisomerase-II (DNA topoisomerases, type II) activity and is used as an antineoplastic and antithumor agent. Experimentally, it has been shown to induce G2 phase arrest in human and murine cell lines. Additionally, genistein has antihelminthic activity. It has been determined to be the active ingredient in *Flemingia vestita*, which is a plant traditionally used against worms. It has also been demonstrated to be effective against intestinal parasites such as the common liver fluke, pork trematode and poultry cestode. [Wikipedia] Further, genistein is a phytoestrogen which has selective estrogen receptor modulator properties. It has been investigated in clinical trials as an alternative to classical hormone therapy to help prevent cardiovascular disease in postmenopausal women [1]. Genistein can be found in food sources such as tofu, fava beans, soybeans, kudzu, and lupin. It is also present in certain cell cultures and medicinal plants. [Wikipedia]

The "Structure" section shows the chemical structure of Genistein (4',5,7-Trihydroxyisoflavone), represented as a 2D diagram. Below the structure are buttons for "View 3D Structure" and file formats: "MOL", "SDF", "3D-SDF", "PDB", "SMILES", and "InChI".

The "Synonyms" section lists: "4',5,7-Trihydroxyisoflavone" and "5,7,4'-Trihydroxyisoflavone". A "Activate Windows" watermark is visible in the bottom right corner.

Fig. 9 Drug bank web page for identification of a drug with ID

4 Conclusion

This paper investigates the databases of protein and gene, sequence alignment, docking and drug discovery which indicated that for various tasks and the performance of few operations. It is useful to understand not only the protein performance, but also the biological activities of the various genes. Through this paper, types of sequence alignment, steps of BLAST and FASTA operation were to be utilized. Docking helps to know the information about the binding site of the protein, and drug discovery uses to understand the steps of drug development processes. From here, the overview of some experiments and sequential steps of some biological tools, databases, sequences, docking, and drug discovery is to be discussed.

References

1. Smith TF, Waterman MS (1991) Identification of common molecular sequences. *J Mol Biol* 147:195–197
2. Needleman SB, Wunsch CD (1990) A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol* 48:443–453
3. Thompson JD, Higgins DG, Gibson TJ (1994) Improved sensitivity of profile searched through the use of sequence weights and gap excision. *CABIOS* 10:19–29
4. Morgenstern B (1999) Dialign, 2 improvement of the segment-to-segment approach to multiple sequence alignment. *Bioinformatics* 15:211–218
5. Thompson JD, Linard B, Lecompte D, Poch O (2011) A comprehensive benchmark study of multiple sequence alignment methods: current challenges and future perspectives. *PLoS ONE* 6:1–14
6. Lassmann T, Sonnhammer EL (2005) Kalign—an accurate and fast multiple sequence alignment algorithm. *BMC Bioinf* 6:298
7. Mount DW (2004) Bioinformatics: sequence and genome analysis. Cold Spring Harbor Laboratory Press, Cold Spring Harbor, NY
8. Altschul SF, Madden TL, Schaffer AA, Zhang J, Zhang Z, Miller W et al (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res* 25(17):3389–3402 (PMC free article) (PubMed)
9. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ (1990) Basic local alignment search tool. *J Mol Biol* 215(3):403–410 (PubMed)
10. Baker EJ, Lin GN, Liu H, Kosuri R (2007) NFU-enabled FASTA: moving bioinformatics applications onto wide area networks. *Biol Med* 2. PMC2211279
11. Chen R, Li L, Weng Z (2003) ZDOCK: an initial-stage protein-docking algorithm. *Proteins* 52:80–87
12. Song CM, Lim SJ, Tong JC (2009) Recent advances in computer-aided drug design. *Brief Bioinform* 10:579–591
13. Worrell J (2006) Practical issues of conducting clinical trials in India. *Drug Dev*

Machine Learning for Bioinformatics



K. Aditya Shastry and H. A. Sanjay

1 Introduction

Bioinformatics represents an interdisciplinary branch for developing improved methods for retrieving, analyzing, storing, and organizing the biological data. It focuses on the development of algorithms and software for the transfer, storage, analysis, and development of genomics databases [1].

The understanding of the gene functionality along with cell regulations, selection of drug targets, design of drugs, and disease identification is handled by the bioinformatics approaches. The interpretation of enormous amounts of biological data generated by several biological systems, and genomics requires quantitative analysis. There are requirements for skilled bioinformaticians in both industry and academia for developing novel data management and analysis methods [2].

Enormous amounts of biological data are being generated due to the rapid developments taking place in computing hardware and software and high throughput technologies in genomics, systems biology, and deep-sequencing. For the analysis of this big biological data, novel approaches in bioinformatics are required. This has led to tremendous opportunities being created for bioinformatics scientists in order to solve these new issues and challenges [3].

Machine learning (ML) belongs to the branch of computer science that provides self-learning capability to the machines without explicit programming. The ML algorithms are being extensively used for the tasks of prediction, classification, and feature selection in bioinformatics. The ML approaches are very good for solving problems such as distinguishing between DNA sequences and classification of DNA sequences. Currently, the ML in bioinformatics has become significant due to the advent of deep learning [4].

K. A. Shastry (✉) · H. A. Sanjay

Department of Information Science and Engineering, Nitte Meenakshi Institute of Technology, Bangalore, India

e-mail: adityashastry.k@nmit.ac.in

The ML algorithms are used for selecting relevant features in bioinformatics application as biological data are high dimensional in nature [5]. Some of the recent works in feature selection using ML for bioinformatics is summarized below:

- Liu et al. [6] developed a double RBF kernel method for selecting relevant features from gene expression data for cancer classification. The irrelevant and redundant genes are removed by combining RBF kernels using weighted analysis and relevant feature genes are extracted.
- The authors Masoudi-Sobhanzadeh et al. [7] have developed a tool called FeatureSelect consisting of optimization algorithms and learners for gene feature selection.
- Relief machine learning algorithms like nearest neighbors by Le [8] was used to select the features on relief based scores. The statistical importance of features was computed.

The second major application of ML in bioinformatics has been found in classification of biological data. Some of the recent works in classification of biological data using machine learning techniques is highlighted below:

- Budach et al. [9] have used convolution neural networks (CNN) to classify biological sequences by learning sequence and structure motifs.
- In [10], the authors have utilized deep learning approach (convolution neural networks and deep belief networks) for the taxonomic classification of metagenomic data. The CNN had 10 kernels in first layer with size 5 and 20 kernels in second layer.

The third major application of ML in bioinformatics was found to be prediction of biological data. Some of the current works related to prediction are described below:

- Tsubaki et al. [11] have predicted the compound-protein interaction (CPI) by combining graph neural network [GNN] and CNN. The accurate CPI prediction assists in effective drug discovery.
- Karimi et al. [12] have used the unified recurrent and CNN for accurate CPI prediction.
- In the work [13], the authors have performed protein function prediction using fusion of deep neural networks based on multi-modal deep encoders for mining the features of proteins which are high-level. They are extracted from several interaction networks that are heterogeneous in nature.

The fourth major ML application in bioinformatics is clustering of biological data like gene data. Similar genes are clustered together. Following points highlight some recent works in the area of gene clustering using ML:

- In [14], authors have designed a Cluster Locator tool which demonstrates the distribution of genes and gene clusters with respect to a reference genome.
- The k-windows clustering algorithm was used by the authors Tasoulis et al. [15] on gene expression microarray data.

- The work by Zhang et al. [16] developed a clustering framework called DendroSplit. The DendroSplit generates clusters using hierarchical clustering based on complete link method.

The rapid growth of biological data has raised two issues: First issue is the storage of information in an efficient manner, and the second issue is the mining of knowledge which is useful from the data. The second issue is the foremost challenge in computational biology as it requires tool development and techniques which can transform the heterogeneous data into biological knowledge. These tools help in providing knowledge using models which are testable which in turn enables prediction of the system [17].

The application of ML methods has been observed in several biological domains such as microarrays, proteomics, genomics, systems biology, text mining, and evolution for mining useful information from the biological data [18].

Genomics represents a very significant field in bioinformatics. The genomic sequence increases exponentially. The processing of this sequence is needed to acquire useful information. Initially, the gene structure and its location from genomic sequence are extracted. The prediction of secondary structure and gene function is performed using sequence information [19].

The proteins convert the genetic information into life. The 3D configuration of a protein plays a significant role in its functioning process. The forecast of protein structure is a complex process which involves several atoms, bounds, and optimization. Hence, ML techniques are useful for this purpose [20].

The management of complex experimental data forms a fascinating area of research in the area of bioinformatics. This type of data which is collected is represented by a domain called microarray essay. Two common problems occur when dealing with complex experimental data. Firstly, the preprocessing of data needs to be done. Secondly, the proper data analysis needs to be performed. For microarray data, identification of expression patterns and genetic network induction are the most common applications [21].

The ML techniques are very useful in the domain of systems biology as the modeling of life processes within a cell is very complex. Hence, the computational ML methods are particularly useful for modeling biological networks such as metabolic pathways, genetic networks, and signal transduction networks [22].

The creation of phylogenetic tree along with evolution can be implemented using ML techniques. The evolution of organisms is schematically represented by the phylogenetic trees. Previously, the trees were created utilizing the diverse features such as morphological, metabolic features. However, in recent days due to the availability of genome sequences the comparison of different genomes is utilized by the phylogenetic tree construction algorithms. Multiple sequence alignments are used where the different optimization techniques are beneficial [23].

The increasing amount of publications in bioinformatics using ML has led to the advent of text mining. The text mining methods may be utilized to extract knowledge from these publications. It is being used in functional annotation, analysis of protein interaction, and the prediction of cellular locations [24].

Other problems related to the efficient design of primer for PCR, analysis of biological images, and back translation of proteins are being solved by machine learning techniques. Machine learning involves the automatic learning of computers without explicit programming. It usually learns through past experience. The term learning refers to the concept where a computer program is able to learn by itself from training data. ML utilizes statistical theory for building the models by deriving inferences from a sample [24].

The ML process constitutes of two main steps. The first step is to produce a model by analyzing the big data. The second step involves deriving inferences from the analysis. Apart from the algorithm efficiency, other factors such as time and space complexity, transparency and interpretability are found to be important for accurately predicting the results. Extracting useful knowledge from data is an iterative and interactive process. The iterative stage is composed of many steps. In the first step, the data from different sources is merged into a single format. Data-warehouse techniques can be used for identifying the outliers and inconsistencies. The second step involves the selection, cleaning, and transformation of the collected data [24].

2 Machine Learning Techniques in Bioinformatics

The underlying notion of applying ML techniques in bioinformatics is to extract or find out useful information from the biological databases. This information needs to be presented to the end user in a meaningful and understandable pattern. The ML tasks in bioinformatics involve classification, clustering, prediction, link analysis (finding associations), finding a group, detection of deviation (finding changes), and visualization (presenting data visually for easy human interpretation) [24].

The Knowledge Discovery (KDD) Process in a biomolecular database is demonstrated in Fig. 1.

The pattern represents the regular expression that is understandable by humans. The critical KDD features are the construction of languages of patterns (high-level) which are comprehensible by humans accurately. Novel and useful knowledge are produced by the patterns that require less complexity and computability.

2.1 Artificial Neural Network (ANN) in Bioinformatics

The ANN is used widely in bioinformatics as it was found to be efficient during the early stages of research in bioinformatics [25]. It is able to solve complex real-world problems while being flexible. They are resistant to noise and errors present in the training data. The major disadvantage of ANN is that it is a black-box approach where the functioning of each node in the neural network is hard to interpret and validate. It also involves complex mathematical and statistical computations which

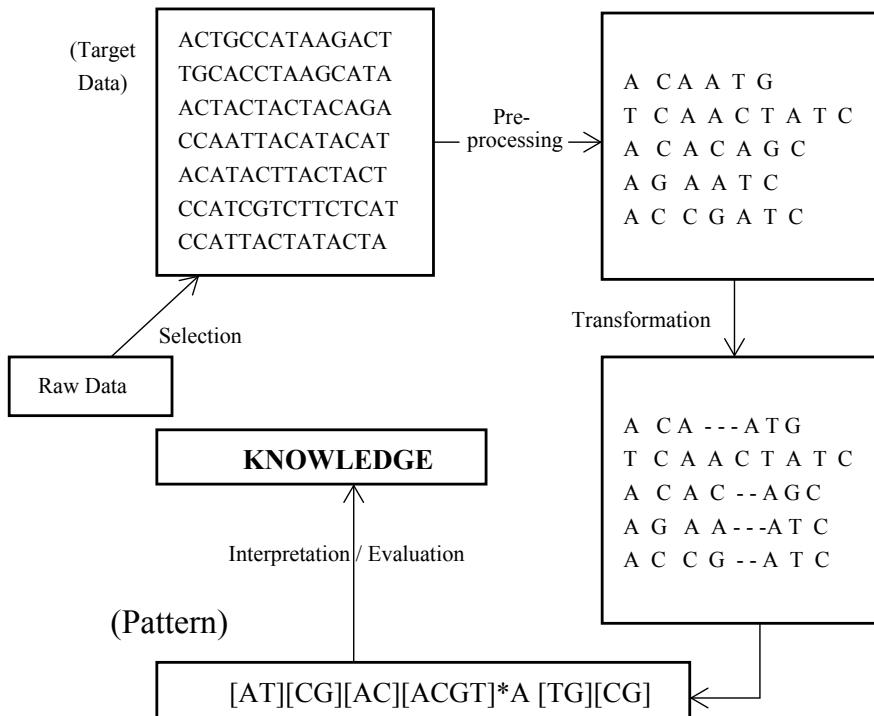


Fig. 1 KDD process in biomolecular database [24]

make it hard to interpret. The choice of ANN parameters such as its weights, biases, and number of hidden layers is an ongoing research issue [26].

The ANNs have found widespread applications in the prediction of protein structure, functional prediction [27–31], and classification of proteins [32–34].

2.2 Decision Trees in Bioinformatics

The decision tree method has been utilized in the works [35, 36] for classifying membrane protein sequences based on its functional classes. The prediction of protein structure was performed by He et al. [37, 38] and Sivan et al. [39] using the decision trees. Furthermore, the decision trees were used by Salzberg [40] to trace the protein-coding genes.

For practical applications, the decision trees are relatively better as they are resistant to outliers and noise. However, they tend to over fit in some cases. Optimizing decision trees is difficult in certain instances [41].

2.3 Genetic Algorithms (GA) in Bioinformatics

The genetic algorithms have found popularity in bioinformatics research due to its simplicity. It is more suited to bioinformatics since most of the biological data have high dimensions, and GA is more able to solve high dimensional problems. The major drawback of GA is that the changes during its evolution process are dynamic and not clear [42].

Parsons et al. [43], Rathee et al. [44], Alba et al. [45], and Nebro et al. [46] have applied GA for DNA fragment assembly. Horng et al. [47] used GA for the alignment of multiple molecular sequences.

3 Applying Artificial Neural Network in Bioinformatics: A Case Study

Widespread application of bioinformatics is found in machine learning. ANN is the widely applied machine learning algorithm in bioinformatics. The case study of ANN in bioinformatics is presented in this section [48]. Diverse problems that are considered for bioinformatics mainly fall into the key tasks listed below:

- The task of comparing and aligning the sequences related to RNA, protein, and DNA.
- The task of identification of promoters and finding of genes from sequences related to DNA.
- The task of interpreting the expression-gene and micro-array data.
- The task of identifying the network (regulatory) of genes.
- The task of learning the evolutionary relationship by constructing the phylogenetic trees.
- The task of classifying and predicting the structure of proteins.
- The task of molecular design and docking.

Consequently, the goals of bioinformatics are:

- Data arrangement related to the production and access of information for the benefit of researchers.
- Tool development to simplify the data management and its analysis.
- To make the result interpretation and analysis of biological data more comprehensible.

Several algorithms are developed to accomplish the objectives stated above. Certain trends in algorithms that are observed in bioinformatics are summarized below:

- Discovering similarities between strings (like organisms, DNA, or proteins).
- Identifying specific patterns present in strings (like genes, α -helices, and introns).
- Detecting similarities between portions of spatial structures (like motifs).

- For creation of phylogenetic trees.
- Categorizing new data based on earlier clustered sets of annotated data.
- Analysis of micro-array data.

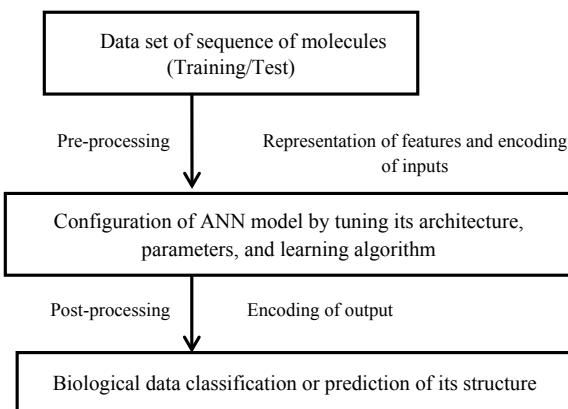
3.1 Designing ANN for Bioinformatics

The common issues faced while designing ANN for bioinformatics applications need to be addressed. Figure 2 demonstrates the flowchart of ANN design for bioinformatics.

The feature representation and input encoding constitute the data preprocessing stage. The performance of ANN is determined by the proper data preprocessing steps. The prior knowledge about the structure and function of sequences has to be represented to obtain the complete benefit of ANN. This enables the effective mining of relevant attributes present in the sequence. Different combinations of twenty amino acids with different lengths make up the proteins. Defined alphabet set is used to write a sequence. However, in biology, the alphabets carry additional information such as function and residue structure. The amino acid chain reveals a great number of physical and chemical properties. The structure and function of protein are determined by the interaction between the side chains. Prior knowledge encoding of certain amino acid properties such as surface area, volume, propensity of secondary structure, polarity, and chemical properties is performed when the ANN is fed with input [49].

Certain amino acids demonstrate multiple structural or functional roles; hence, their properties are assessed by considering the entire protein. These properties represent global or local features of protein context. To characterize these attributes, hydrophobic moments, amino acid frequency, or hydrophobicity profile may be computed and encoded to use in ANN.

Fig. 2 ANN design for bioinformatics [49]



The evolutionary feature of protein is signified as replacement metrics such as BLOSUM or PAM. ANN uses this information after encoding. Once the features are identified for representing in the ANN application, several methods are utilized to characterize these data to take full advantage of information extraction. They are highlighted below:

- Measuring the real number with respect to continuous scale like Mass.
- Frequencies or vectors representing distances viz. PAM matrix.
- Class categorization.
- Utilizing alphabet alternatively to denote AA possessing attributes that are similar.
- Class that is hierarchical in nature.

Data is encoded after the identification of attributes which require to be characterized in the ANN application. Local or global encoding may be done. In local encoding, the short sequence segments comprising of single or neighboring residues are performed. The global encoding involves long-range relationship in the entire sequence. Direct or indirect method is present for sequence encoding. The positional information is preserved, and each residue is transformed into a vector in direct encoding. Indirect encoding is utilized to provide the information measure as a whole for the complete sequence.

Output encoding is simple when compared to input encoding. It is reliant on number of classifications required in application. The number of output units is automatically configured in neural networks like self-organizing maps. The output unit values are used as a qualitative or quantitative measure of activity or confidence level [50].

3.2 ANN in Protein Bioinformatics

The ANN finds its use in several bioinformatics applications involving proteins. They can be categorized into the following categories:

- Protein structure prediction containing content of the secondary structure, structural contacts, maps of contacts, domain of boundaries related to structures, beta-turns, etc.
- Forecast of ligands and binding sites that comprises of forecasting the binding residues and prediction of several properties of the binding ligands.
- Forecast of the properties of proteins like proteins related to physicochemical, host organism localization, etc.

The forecast of the protein secondary structure using ANN is presented in this section. The PSIPRED is the commonly used model for the forecast of protein secondary structure. It was proposed by Jones in 1999. The PSIPRED method is a two-stage ANN. It takes a position specific scoring matrix (PSSM) that is created using PSI-BLAST algorithm by way of the input. Figure 3 depicts the design of PSIPRED.

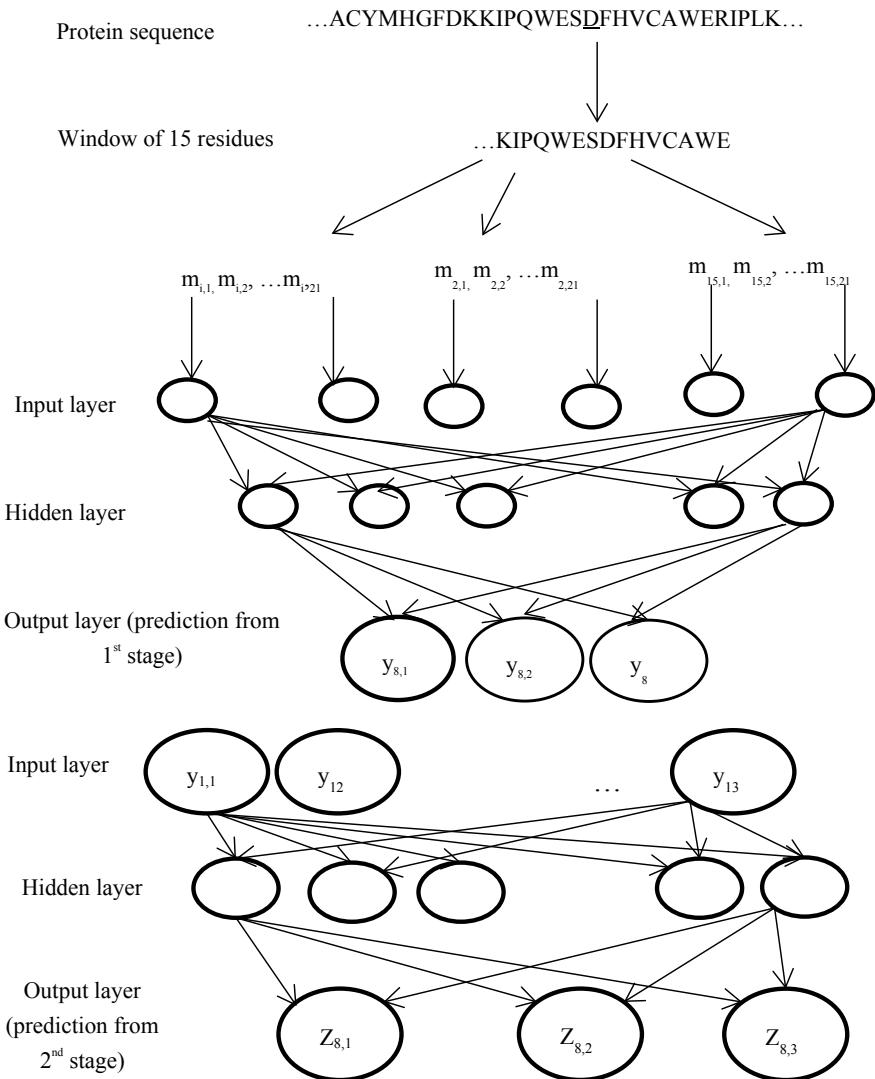


Fig. 3 Architecture of PSIPRED algorithm [31]

Figure 3 illustrates a two-stage network that has three layers of feed-forward networks. Here, the input to the second stage network is the output of the first stage network. During the first stage, the PSSM is used to represent the input protein sequence utilizing a window size of 15 which is based on the predicted amino acid. Twenty dimensions are included in the PSSM for each amino acid with substitution scores. The scores are used to indicate the presence/absence of amino acid at a particular position in the sequence from a set of sequences which resemble the predicted

sequence. Similarity in the structure is based on the similarity in the sequence. A positive score specifies the frequent occurrence of a particular amino acid. Similarly, negative score specifies the infrequent occurrence of a particular amino acid. During the first stage, the input layer of the ANN is fed with twenty PSSM scores with a feature indicating the boundary of the sequence. Consequently, $15 \times 21 = 315$ nodes are present in the input layer. The hidden layer is composed of 75 nodes and three nodes indicating the probability of three secondary structure states form the output layer [51].

The input layer of the second stage is fed with the output of the first stage which is represented by the probabilities that were predicted for the secondary structures for a window of 15 amino acids. The structure of the amino acids in the positions that are adjacent to each other in the sequence is determined by the second layer. The input layer is comprised of $4 \times 15 = 60$ nodes, hidden layer is comprised of 60 nodes, and three nodes are contained in the output layer. The URL: <http://bioinf.cs.ucl.ac.uk/psipred/> gives the access to the PSIPRED method. The stand-alone version of this technique can be downloaded by interested users [51].

Cascaded bidirectional recurrent neural network (BRNN) is recently being used to perform secondary structure prediction. As in PSIPRED design, during the first stage, the BRNN forecasts the secondary structure from the input amino acid sequences. In the second stage, the raw predictions from first stage are refined. Back-propagation is used as the learning algorithm.

The previous two decades have seen the development of many NN-based methods for forecasting the protein secondary structure. The representation of the protein sequence along with training set size determines the performance of the methods. The window-based methods for the estimation of protein structure are ineffective in capturing the long-range reactions since the beta sheets are created between the amino acids that are distant in the sequence. This leads to poor results for strands [31].

4 Research Issues Related to Machine Learning in Bioinformatics

Various research issues or challenges of machine learning methods in bioinformatics are discussed in this section.

4.1 Data Errors

Presence of duplicate data is a significant issue in bioinformatics. With the advent of Internet, the data is publicly available which makes it difficult to detect errors and measure the data quality.

The data source was not considered by the bioinformatics researchers during the utilization of the data. If the data at source is dirty, then the accuracy of the classifier decreases. Dirty biological data can be due to many factors such as:

- i. Errors during experimentation.
- ii. Erroneous interpretation by biologists.
- iii. Typing mistakes due to human error.
- iv. Non-standardized methods (3D structure in PDB from multiple sources, X-ray diffraction, theoretical modeling, nuclear magnetic resonance, etc.) are used in experiments.

The learning algorithms need to be robust to data when learning on a biological database that is dirty. The ML approaches must be able to provide optimal decisions by adjusting in order to avoid over fitting the datasets. Since the biological databases are updated on a daily basis, the ML algorithms should possess short learning and training times [24].

While interpreting and analyzing biological, it is essential to work with domain experts which in this case are biologists. To maintain the data quality, the data in the databases needs to be constantly revised. The issues faced during the cleaning of gene databases are addressed by Brunak et al. [52] and Korning et al. [53]. Applying the ML techniques on biological databases to learn has developed into a major challenge in bioinformatics research.

4.2 Generative Versus Discriminative

Most of the hypotheses in biological research are data driven and based on experimental data. Further experimental-based research suffers from uncertainty. Hence, the biological research has been transformed into a statistical dependent field. Without the incorporation of probability theories, the degree of confidence will be lower when bioinformatics analysis is performed. To differentiate between classes, a discriminative method is useful, while a general approach is required in other cases. The choice of ML techniques in bioinformatics is dependent on the objectives of learning and application tasks. More confidence is provided to the discovered knowledge if the appropriate method is implemented in bioinformatics research. This, in turn, improves the degree of uncertainty [24].

4.3 Approximation Versus Explanation

Approximation and explanation form the two issues in bioinformatics community. Certain machine learning methods like ANN, genetic algorithms, etc. generate outputs from the learning process without explaining the underlying concepts [54–56]. These methods are often known as ‘black box’ approaches in ML since they are hard

to understand and interpret. However, they generate better results than the inductive learning approaches that are rule based. Traditional rule generators like decision trees provide clear explanations understandable by humans. This is an open research topic where we have to decide whether we need a black-box approach or an approach with clear explanation at each step [24].

4.4 Single Versus Multiple Methods

Nowadays, hybrid approaches are finding popularity in bioinformatics research. This is because when several methods are combined, the limitation of each method gets nullified and benefits of each method get added. This results in a better performance [24].

5 Conclusion

Recently, the transformation of huge volume of data into knowledge is the biggest challenge faced in computational biology. The machine learning techniques provide this transformation. This chapter introduces some useful ML methods like genetic algorithms, decision trees, and artificial neural networks. From literature, it is observed that artificial neural networks are the widely used technique in bioinformatics research. Keeping this mind, this chapter discusses in detail the case study of protein structure prediction using ANN. Some of the critical research issues in bioinformatics are also discussed.

References

1. Abdurakhmonov IY (2016) Bioinformatics: basics, development, and future. IntechOpen. <http://dx.doi.org/10.5772/63817>
2. Hakeem K, Mujtaba Babar M, Sadaf Zaidi N-u-S, Pothineni V, Ali Z, Faisal S, Gul A (2017) Application of bioinformatics and system biology in medicinal plant studies. https://doi.org/10.1007/978-3-319-67156-7_15
3. Yin Z, Lan H, Tan G, Lu M, Vasilakos AV, Liu W (2017) Computing platforms for big biological data analytics: perspectives and challenges. Comput Struct Biotechnol J 15:403–411. ISSN 2001-0370. <https://doi.org/10.1016/j.csbj.2017.07.004>
4. Awad M, Khanna R (2015) Machine learning. Efficient learning machines. Apress, Berkeley, CA
5. Saeys Y, Inza I, Larrañaga P (2007) A review of feature selection techniques in bioinformatics. Bioinformatics 23(19):2507–2517. <https://doi.org/10.1093/bioinformatics/btm344>
6. Liu S, Xu C, Zhang Y, Liu J, Yu B, Liu X, Dehmer M (2018) Feature selection of gene expression data for Cancer classification using double RBF-kernels. BMC Bioinform 19(1):396. <https://doi.org/10.1186/s12859-018-2400-2>

7. Masoudi-Sobhanzadeh Y, Motieghader H, Masoudi-Nejad A (2019) FeatureSelect: a software for feature selection based on machine learning approaches. BMC Bioinform 20:170. <https://doi.org/10.1186/s12859-019-2754-0>
8. Le T, Urbanowicz R, Moore J, Mckinney B (2018) STatistical Inference Relief (STIR) feature selection. Bioinformatics (Oxford, England) 35. <https://doi.org/10.1093/bioinformatics/bty788>
9. Budach S, Marsico A (2018) pysster: classification of biological sequences by learning sequence and structure motifs with convolutional neural networks. Bioinformatics 34(17):3035–3037. <https://doi.org/10.1093/bioinformatics/bty222>
10. Fiannaca A, La Paglia L, La Rosa M, Lo Bosco G, Renda G, Rizzo R, Gaglio S, Urso A (2018) Deep learning models for bacteria taxonomic classification of metagenomic data. BMC Bioinform 19(Suppl 7):198. <https://doi.org/10.1186/s12859-018-2182-6>. PubMed PMID: 30066629. PMCID: PMC6069770
11. Tsubaki M, Tomii K, Sese J (2019) Compound–protein interaction prediction with end-to-end learning of neural networks for graphs and sequences. Bioinformatics 35(2):309–318. <https://doi.org/10.1093/bioinformatics/bty535>
12. Karimi M, Wu D, Wang Z, Shen Y (2018) DeepAffinity: interpretable deep learning of compound protein affinity through unified recurrent and convolutional neural networks. <https://doi.org/10.1101/351601>
13. Gligorijevic V, Barot M, Bonneau R (2018) deepNF: deep network fusion for protein function prediction. Bioinformatics (Oxford, England) 34. <https://doi.org/10.1093/bioinformatics/bty440>
14. Pazos Obregón F, Soto P, Lavín JL, Cortázar AR, Barrio R, Aransay AM, Cantera R (2018) Cluster Locator, online analysis and visualization of gene clustering. Bioinformatics 34(19):3377–3379. <https://doi.org/10.1093/bioinformatics/bty336>
15. Tasoulis DK, Plagianakos VP, Vrahatis M (2004) Unsupervised clustering of bioinformatics data
16. Zhang J, Fan J, Christina Fan H, Rosenfeld D, Tse DN (2018) An interpretable framework for clustering single-cell RNA-Seq datasets. BMC Bioinform 19. <https://doi.org/10.1186/s12859-018-2092-7>
17. Larranaga P (2006) Machine learning in bioinformatics. Brief Bioinform 7:86–112. <https://doi.org/10.1093/bib/bbk007>
18. Chen Yi-Ping Phoebe (2005) Bioinformatics technologies. Springer, Berlin, Heidelberg
19. Sung W (2012) Bioinformatics applications in genomics. Computer 45(6):57–63. <https://doi.org/10.1109/MC.2012.151>
20. Rokde CN, Kshirsagar M (2013) Bioinformatics: protein structure prediction. In: 2013 fourth international conference on computing, communications and networking technologies (ICCCNT), Tiruchengode, pp 1–5. <https://doi.org/10.1109/icccnt.2013.6726753>
21. Moreau Y, De Smet F, Thijs G, Marchal K, De Moor B (2002) Functional bioinformatics of microarray data: from expression to regulation. Proc IEEE 90(11):1722–1743. <https://doi.org/10.1109/JPROC.2002.804681>
22. Yeol JW, Barjis I, Ryu YS (2005) Modeling of system biology: from DNA to protein by automata networks. In: Proceedings of 2005 international conference on intelligent sensing and information processing, Chennai, India, 2005, pp 523–528. <https://doi.org/10.1109/icisip.2005.1529510>
23. Bereg S, Bean K (2005) Constructing phylogenetic networks from trees. In: Fifth IEEE symposium on bioinformatics and bioengineering (BIBE'05), Minneapolis, MN, USA, pp 299–305. <https://doi.org/10.1109/bibe.2005.19>
24. Tan AC, Gilbert D (2001) Machine learning and its application to bioinformatics: an overview
25. Stormo G, Schneider T, Gold L, Ehrenfeucht A (1982) Use of the perceptron algorithm to distinguish translational initiation in *E. coli*. Nucleic Acids Res 10:2997–3011
26. Li Y, Huang C, Ding L, Li Z, Pan Y, Gao X (2019) Deep learning in bioinformatics: introduction, application, and perspective in the big data era. Methods. ISSN 1046-2023. <https://doi.org/10.1016/j.ymeth.2019.04.008>

27. Hirst JD, Sternberg MJE (1992) Prediction of structural and functional features of protein and nucleic acid sequences by artificial neural networks. *Biochemistry* 31:7211–7218
28. Qian N, Sejnowski TJ (1988) Predicting the secondary structure of globular proteins using neural network models. *J Mol Biol* 202:865–884
29. Howard Holley L, Karplus M (1989) Protein secondary structure prediction with a neural network. *Proc Natl Acad Sci USA* 86:152–156
30. Mathkour H, Ahmad M (2010) An integrated approach for protein structure prediction using artificial neural network. In: International conference on computer engineering and applications, vol 2, pp 484–488. <https://doi.org/10.1109/ICCEA.2010.243>
31. Chen K, Kurgan LA (2012) Neural networks in bioinformatics. In: Rozenberg G, Bäck T, Kok JN (eds) *Handbook of natural computing*. Springer, Berlin, Heidelberg
32. Rossi ALD, de Oliveira Camargo-Brunetto MA (2007) Protein classification using artificial neural networks with different protein encoding methods. In: Seventh international conference on intelligent systems design and applications (ISDA 2007), Rio de Janeiro, pp 169–176. <https://doi.org/10.1109/isda.2007.81>
33. Rossi A, Camargo-Brunetto MA (2007) Protein classification using artificial neural networks with different protein encoding methods. <https://doi.org/10.1109/isda.2007.81>
34. Lee NK, Wang D, Wah Tan K (2005) Protein classification using neural networks: a review
35. Nijil RN, Mahalekshmi T (2018) Multilabel classification of membrane protein in human by decision tree (DT) approach. *Biomed Pharmacol J* 11(1)
36. Siva Sankari E, Manimegalai D (2017) Predicting membrane protein types using various decision tree classifiers based on various modes of general PseAAC for imbalanced datasets. *J Theor Biol* 435. <https://doi.org/10.1016/j.jtbi.2017.09.018>
37. He J, Hu HJ, Harrison R, Tai PC, Dong Y, Pan Y (2005) Understanding protein structure prediction using SVM_DT. In: Chen G, Pan Y, Guo M, Lu J (eds) *Parallel and distributed processing and applications—ISPA 2005 workshops*. ISPA 2005. Lecture notes in computer science, vol 3759. Springer, Berlin, Heidelberg
38. He J, Hu H-J, Harrison R, Tai PC, Pan Y (2006) Rule generation for protein secondary structure prediction with support vector machines and decision tree. *IEEE Trans Nano Biosci* 5(1):46–53. <https://doi.org/10.1109/TNB.2005.864021>
39. Sivan S, Filo O, Siegelmann H (2007) Application of expert networks for predicting proteins secondary structure. *Biomol Eng* 24:237–243. <https://doi.org/10.1016/j.bioeng.2006.12.001>
40. Salzberg S, Delcher AL, Fasman K, Henderson J (1998) A decision tree system for finding genes in DNA. *J Comput Biol* 5:667–680. <https://doi.org/10.1089/cmb.1998.5.667>
41. Stiglic G, Kocbek S, Pernek I, Kokol P (2012) Comprehensive decision tree models in bioinformatics
42. Bhaskara Murthy V, Pardha Saradhi Varma G (2013) Genetic algorithm—a case study in gene identification. *Int J Adv Res Comput Sci* 4(5)
43. Parsons RJ, Forrest S, Burks C (1995) *Mach Learn* 21:11. <https://doi.org/10.1007/BF00993377>
44. Rathee M, Vijay Kumar TV (2014) DNA fragment assembly using multi-objective genetic algorithms. *Int J Appl Evol Comput* 5(3):84–108
45. Alba E, Luque G, Khuri S (2005) Assembling DNA fragments with parallel algorithms. In: 2005 IEEE congress on evolutionary computation, Edinburgh, Scotland, vol 1, pp 57–64. <https://doi.org/10.1109/cec.2005.1554667>
46. Nebro AJ, Luque G, Luna F, Alba E (2008) DNA fragment assembly using a grid-based genetic algorithm. *Comput Oper Res* 35(9):2776–2790. ISSN 0305-0548. <https://doi.org/10.1016/j.cor.2006.12.011>
47. Horng JT, Wu LC, Lin CM et al (2005) Soft Comput 9:407. <https://doi.org/10.1007/s00500-004-0356-9>
48. Bhaskar H, Hoyle DC, Singh S (2006) Machine learning in bioinformatics: a brief survey and recommendations for practitioners. *Comput Biol Med* 36:1104–1125. <https://doi.org/10.1016/j.comphimed.2005.09.002>
49. Hapudeniya M (2010) Artificial neural networks in bioinformatics. *Sri Lanka J Bio-Med Inform* 1:104–111. <https://doi.org/10.4038/sljbmi.v1i2.1719>

50. Seiffert U, Hammer B, Kaski S, Villmann T (2006) Neural networks and machine learning in bioinformatics-theory and applications. In: European symposium on artificial neural networks, pp 521–532
51. Bordoloi H, Sarma K (2019) Protein structure prediction using artificial neural network
52. Brunak S, Engelbrecht J, Knudsen S (1990) Cleaning up gene databases. Nature 343:123
53. Korning PG, Hebsgaard SM, Rouze P, Brunak S (1996) Cleaning the GenBank *Arabidopsis thaliana* data set. Nucleic Acids Res 24:316–320
54. Sekhar SM, Siddesh GM, Manvi SS, Srinivasa KG (2019) Optimized focused web crawler with natural language processing based relevance measure in bioinformatics web sources. Cybern Inf Technol 19(2):146–158
55. Sekhar M, Sivagnanam R, Matt SG, Manvi SS, Gopalalyengar SK (2019) Identification of essential proteins in yeast using mean weighted average and recursive feature elimination. Recent Patents Comput Sci 12(1):5–10
56. Patil SB, Sekhar SM, Siddesh GM, Manvi SS (2017) A method for predicting essential proteins using gene expression data. In: 2017 international conference on smart technologies for smart nation (SmartTechCon). IEEE, pp 1278–1281

Impact of Machine Learning in Bioinformatics Research



E. Naresh, B. P. Vijaya Kumar, Ayesha and Sahana P. Shankar

1 Introduction

The term bioinformatics was depicted by Paulien Hogeweg and Ben Hesper. Bioinformatics integrates classical features of different fields; those fields include as shown in Fig. 1:

(i) Biology: which plays a salient role in capturing the correct template of the biological data, (ii) Information Engineering: which helps in generation and distribution of the information pertaining to the biological data and also performing analysis on

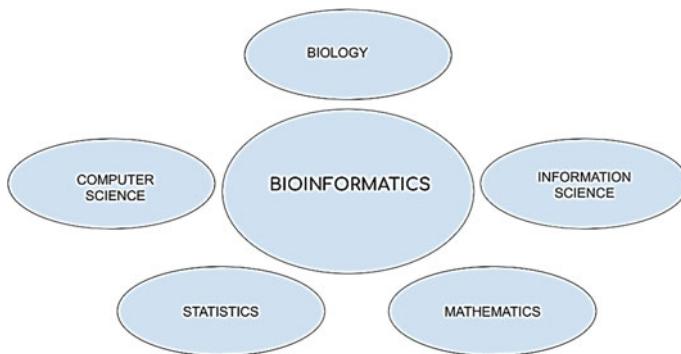


Fig. 1 Fields of bioinformatics

E. Naresh (✉) · B. P. Vijaya Kumar · Ayesha
Department of Information Science and Engineering, M S Ramaiah Institute of Technology,
Bangalore, India

S. P. Shankar
Department of Computer Science and Engineering, Ramaiah University of Applied Sciences,
Bangalore, India

it, (iii) Computer Science: which facilitates the interaction between the processes and the data, (iv) Mathematics and Statistics: which assist in performing all the prerequisite mathematical operation, detailed study, and portraying the biological data. Bioinformatics represents a science of aggregating the biological data, storing the biological data, retrieving, and performing the analysis on the biological data. It also helps in developing software tools in order to entirely have a piece of knowledge about the biological data [1]. The biological data mostly represents genomics, proteomics, microarrays, systems biology, evolution, and text mining.

In recent times, the volume of biological data is enormous coming from various sources of neuroscience. This enormous amount of data needs to be addressed by vigorous data analysis tools and techniques. Although bioinformatics itself serves the purpose, there is a restriction as to the speed of the process and human capacitance and hence we rely on machine learning.

Machine learning consists of significant techniques that aid in bioinformatics research by employing a diverse set of algorithms for data analysis and predictions [1]. Machine learning basically feeds the biological dataset into the system and trains the system on how to reliably distinguish between the dataset and make predictions. These machine learning algorithms help the system learn from the previously recorded biological datasets which can be a genomic dataset or proteomic dataset and thereby help formulate classifiers and hypothesis that unravels complex relation between the biological data. Machine learning can be broadly classified into supervised learning, unsupervised learning, a mixture of both, i.e., semi-supervised, and the last one reinforced learning. A typical machine learning model is depicted in Fig. 2.

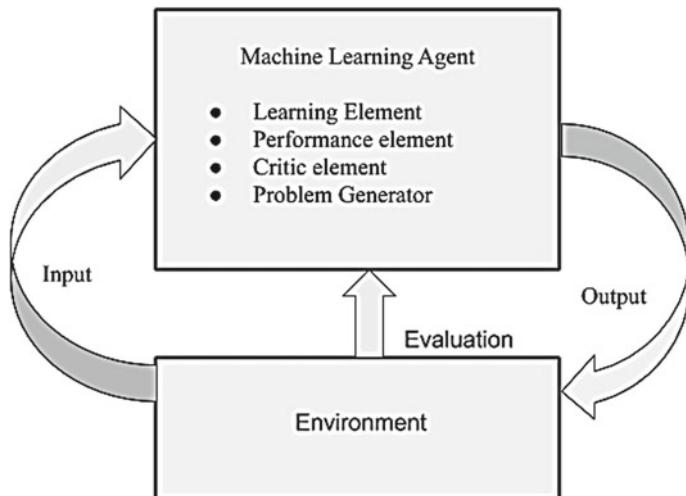


Fig. 2 Typical machine learning model

The biological data such as the genomic data, for example, is a sequence of chemical letters A, C, G, T. The combination of which results in our DNA. Around 99% of our DNA is entirely the same, and the other 1% is what makes us unique. This 1% of unique genome sequence accounts for either uniqueness or some dysfunctionalities in a human. The defect in the genomic sequences is encountered by taking into comparison with the other genomic dataset.

In supervised learning, the system is exposed to a large amount of biological datasets along with their expected labels. This biological data is characterized by the set of attributes that represent them. The set of attributes are the feature set of the object, i.e., a biological dataset which includes genomic sequence from a tissue sample, presence of amino acid at a distinct position, or maybe the absence of it. The main objective of supervised learning is to draft out a model that can accurately predict which class does the new object with its respective feature set belongs to.

Unsupervised learning, on the other hand, is also exposed to an enormous amount of dataset, but there is an absence of the labeled data. Therefore, the system learns about the data which find an intrinsic pattern and cluster them in accordance with their similarities. And it further makes a prediction when it encounters a new dataset by learning and improvising frequently. In a broader sense, the objective behind unsupervised learning is to group the biological datasets by its similarities and define labels to which the dataset is associated.

2 Evolution of Research in Bioinformatics

We may be in an illusion that bioinformatics emerged recently, and the application of machine learning is going to favor in leveling up next-generation data sequence analysis. But bioinformatics was unfolded more than 50 years prior. And in addition to that, the DNA sequencing, in fact, was an unfinished work. Subsequently, the 1960s accounted for the genesis of bioinformatics that involved computational method to deal with protein genomic sequence analysis. Human DNA genome analysis also started to become more observable down the line, due to the improvement in molecular biology and the growth in computer science which facilitated DNA manipulation and innovative software tools helped in better handling of the data, respectively. In due course, however, the biological data saw a tremendous growth thanks to the improvements in sequencing technologies [2]. Hence, traditional database could not account to handle such enormous data; as a result, big data provided ways to tackle huge and complex data in order to address this issue.

2.1 *The Inception of Biological Database*

The initial database was established right after the protein genome sequence was discovered. Subsequently, the nucleic genome sequence was announced after a decade.

Margaret Oakley Dayhoff, a chemist, and a bioinformatics pioneer collected all the recorded genome sequenced data to construct the first-ever bioinformatics database, followed by the Protein Data Bank in 1972 and SwissProt in the year 1987. The structure of these original bioinformatics databases was modest. The data was organized in flat files, and the data entry was mutable. In order to ease, the searching process suitable keywords were used as lookup indices [3].

A biological database is a set of huge persistent data that can be queried, updated, retrieved, and stored [4]. Each record of a nucleotide file per database may normally be associated with input sequence, molecule description, the scientific name of source organism, etc. These biological databases may be publically accessible or some may be private to the organization.

The most tenet requirement biological database is as follows:

- Data quality of the biological dataset needs authentic.
- The database should support relevant experimental data.
- Should have consistent annotation which dictates additional information of the dataset.
- The data should be available whenever it is needed.
- Should allow integration of new experimental biological datasets.

Here are some of the biological database and their description.

2.1.1 GenBank

GenBank is publically accessible repository (Genetic Sequence Databank) which is one of the fastest-growing repositories of known genetic sequences. The dataset contents on the flat files are basically ASCII text files recognized by both humans and systems. It follows an XML data format. It contains information pertaining to the name of the gene, classification of phylogenetic, etc. Recent collection of sequence recorded is shown in the graph in Fig. 3.

2.1.2 EMBL

The EMBL database is of the all-inclusive database which contains DNA and RNA sequences gathered from different sources of submission that includes patent application and scientific research. This EMBL database is in collaboration with the GenBank (set up by the USA) and DNA database which was set up by Japan [3].

2.1.3 SwissProt

SwissProt protein database contains two types of data: the core information and the description of that information (annotation). It allows integration with the other database at a maximum level and maintains a minimum level of redundancies. In

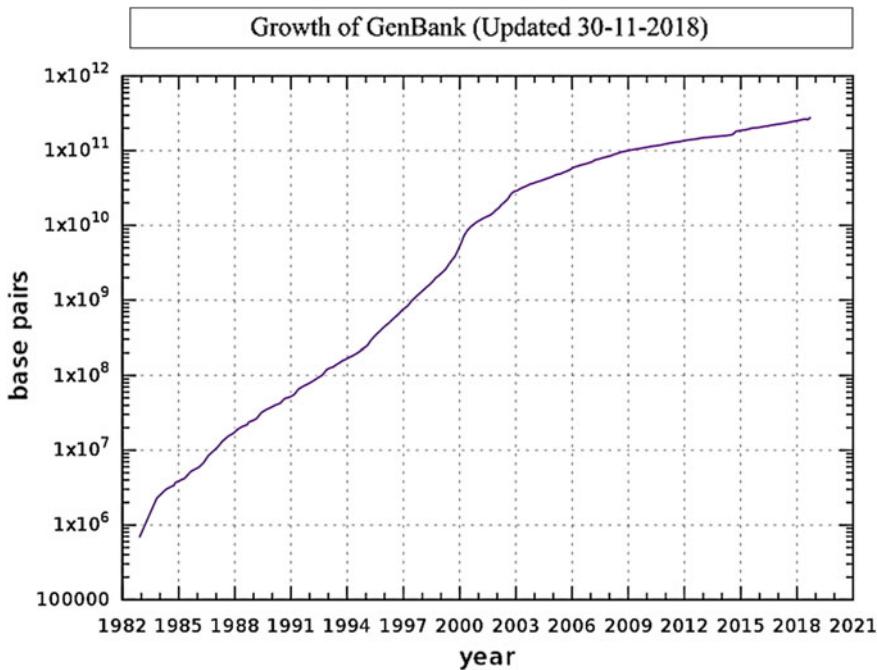


Fig. 3 Genomic data sequence found in GenBank [3]



Participants in the 100,000 Genomes Project are enrolled through any genomic Medicine Center. Their DNA is extracted from blood samples loaded on to a sequencing machine

Fig. 4 Collection phase of the genomic sequence [3]

recent times, the format and the contents of the database were enhanced including cross-references to another integrated database.

2.2 Starting Point: Creation of Sequence Data

The biological database such as the genomic data mostly contains lasting strings of nucleotides that represents guanine, adenine, cytosine, and thymine, and it may or may not contain amino acids such as glycine and serine. Each string sequence of the nucleotide corresponds to the starting letter of the chemical compound that is A, G, C, and T as shown in Fig. 5. The sequence of these letters renders a particular gene or protein. As the chemical composition is only represented by the starting initials,

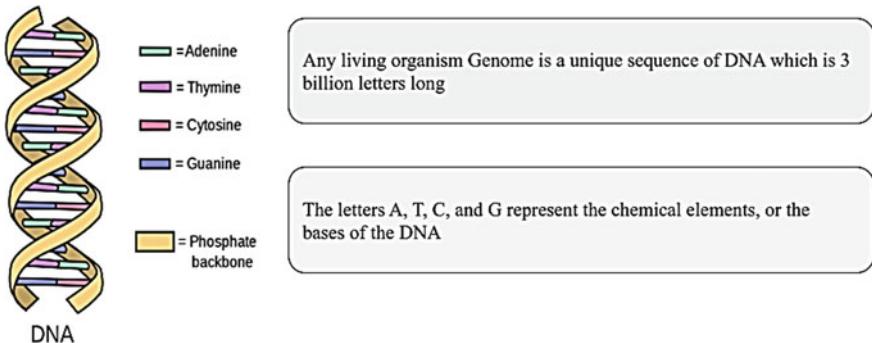


Fig. 5 Chemical composition of a DNA [3, 5]

this alleviates the processing speed and requires less amount of storage for analysis. Most of the database holds information pertaining to the nucleotide sequencing or protein sequencing whereas there are few databases which take core information such as biochemical structures and characteristics of organism into consideration. The advancement in the field of biology and chemistry is speeding up the gene sequencing. Apart from the above advancement, cloning technology also accounted for the production of new DNA sequences which were the outcome of mutation between a foreign DNA and bacteria. Synthesis of the oligonucleotide, on the other hand, provided methods to fabricate small fragments of DNA by choosing other DNA sequences. All these development made it possible to have more and more sequences to deal with and hence increased the progress. The entire procedure from collecting the blood samples to extracting the DNA is depicted in Fig. 4. As the creation of sequences started increasing, accumulating all these sequences from different sources by hand did not seem practical enough. The researchers needed access to the database of sequence information and wanted methods to extract those sequences. After the accumulation of the sequences, the analysis and the structuring was yet to be accomplished.

The computer technology catered the required boost to store and organize the sequence information in the biological database. The computing power and capacity storage were rapidly increasing on all of the above benefits.

2.3 Analysis of Sequence Data

Bioinformatics tools are helpful in acquiring gene sequences and protein sequences from the repository. The repository holds the genomic sequence coming from various sources such as experimental data and research papers of scholars. These tools can also be helpful to analyze different type of sequences which are either labeled or prepared from experiments performed by the researcher [3].



Fig. 6 Analysis of sequences [6]

The analysis begins when an abnormality is observed in a sequence. The test sequences are mapped to the previously recorded sample sequences. Every chemical letter of the sequence in the test sample is aligned and compared with the chemical letter of the recorded sample as illustrated in Fig. 6. The unmatched chemical letter depicts the abnormalities in the sequences. The question arises as to how will the 6 billion chemical letters are compared. Answer to that is 1% of our genomic sequence accounts for the uniqueness. That one 1% of the genomic sequence is basically the test samples which is compared with the recorded sample. The genomic sequences in our body are like a code to the computer. Just like the code gives instruction to the computer, in the same way genomic sequence gives instruction to our bodies. The 1% of the genomic sequence depicts the color of a human's eyes, hair, etc. and also depicts the abnormalities in human. Note that this is in reference to biological terms that differs slightly in computational terms.

3 Case Study BLAST

BLAST is a bioinformatics algorithm that compares genomic sequences of nucleotide or proteins with the available sequences of nucleotide and proteins in the biological database. It permits a researcher to perform analysis on the given test sequence and

help in identifying the congruency between the sequences of the test sample and the recorded sample in the database at certain constraints. The abbreviation of BLAST is a basic local alignment search tool. It is an open-sourced algorithm that works on different operating systems such as Linux, UNIX, Mac, and Windows.

The main components of BLAST are input, output, process, and algorithm.

Input: Inputs are the sequences obtained from the GenBank in the same format as the GenBank and the weight matrix.

Output: The output in BLAST supports HTML format, plain text format, and XML format. The default format is an HTML format for its creators that is NCBI. The results include a graphical representation which indicates sequence hits, tabular representation for the identifiers of the sequence, and sequence alignment with respect to the sequence hits. The sequence is a term that describes the matched sequence. The most instructive output among these is the tabular representation.

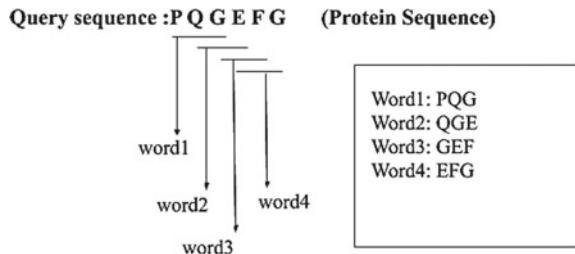
Process: The process under BLAST is to search for similar sequence [7]. It is done by the process called seeding. Seeding is the process of tracing short string matches between the sequences. When the first match is met the BLAST process starts aligning the sequences. In order to find the identical, the algorithm breaks down the sequence of the nucleotide or amino acid into short strings called the seed of minimal size. For example, if the sequence string is GLKFG (sequence of amino acid), short seeds built by the BLAST algorithm are GLK, LKF, and KFG. These seeds are of the size three which is minimal for five letter sequences of the amino acid. The process further continues to compare the respective sequences that are the test sample and the recorded samples, when the short seed is a sequence hit (sequence matched). The process of aligning takes place. The alignment has only accepted the score of the seed corresponds to the threshold T of the scoring matrix. A scoring matrix is a matrix containing optimal sequence similarity. The normally used scoring matrix in BLAST is BLOSUM62. The threshold T decides whether the seeds have to be included or not. If the score of the seed is higher than the threshold, then the seed is aggregated to the alignment. On the other case, if the score of the seed is lower than the threshold, then the seed is rejected and the alignment ceases.

3.1 *BLAST Algorithm*

Step 1: Withdraw low complexity region or repetitive sequences from the sequence of the query.

When few types of element form a sequence, this region is termed as low complexity region. The low complexity region leads to a high score which may cause ambiguity to the program that is programmed to find the substantial sequences. Hence, this low complexity region needs to be eliminated from the biological database. When we consider protein sequence, these regions are tagged with the letter X. And in the case of nucleic sequence, the region is tagged by the letter N.

Fig. 7 k -length comparison of the query sequence [6]



The BLAST program complies and disregards the sequence tagged by the respective letter. SEG and DUST are used to omit the low complexity region of protein and DNA sequences, respectively.

Step 2: Create a minimal sequence of the size k for the query sequence.

Equate letter k to a small constant that represents the length of the sequence. The BLAST algorithm performs a possible combination of the query sequence restricted to the k length. From the beginning of the first letter of the query sequence until it reaches to the end letter of the sequence. Normally, $k = 11$ for the sequence of the DNA.

Step 3: Enumerate viable matches.

BLAST algorithm considers matches if and only if the matches are high scoring words (sequences). The comparison of the k -length sequence with the query sequence gives rise to the score which is recorded in the scoring matrix. The possible match score when $k = 3$ is denoted by 20 raised to k (20^k). When the match scores are recorded, the scores are compared with the threshold score T . If the recorded scores are higher than the threshold T , the recorded scores are accepted and rejected if otherwise (Fig. 7).

Step 4: Structure an efficient search tree that contains high scoring sequence which is resultant from the above step.

The resultant high scoring sequences are now compared to the sequences in the biological database. The biological database's sequence is scanned with high scoring sequences to find its accurate match. When the match is discovered, the seeding process takes place that is nothing much the matched sequence is aligned between the sequence that is queried and sequences from the database. For every k -length sequence, steps 3 and 4 are repeated.

Step 5: Scale the accurate matches to HSP (high scoring segment pair).

In this step, the BLAST algorithm expands the aligned sequences between the database and the query sequence in both the direction of the sequence. This occurs right from that position where the match was found. It does not stop until the high scoring segment pair begins to reduce.

Step 6: Enumerate all the high segment pair in the database having a high enough score.

A cutoff score is fixed experimentally, denoted by the letter S . If the high segment pair is greater than the cutoff value, the HSP makes its way to the list. The cutoff score is fixed by analyzing the alignment distribution score designed by comparing random sequence in such a way that the cutoff score is identified to have highest possible value to assure importance to the rest of the HSP.

Step 7: The importance of HSP is examined.

BLAST algorithm examines the importance of HSP one by one statistically by using extreme value distribution (EVD). It is evident that for local alignment of ungapped sequence that follows Smith-Waterman distribution, Gumble extreme value distribution is applicable perfectly.

In the above accordance, probability of scored observed equal to or greater than x is given by Eq. (1)

$$P(S \geq x) = 1 - \exp(-\lambda(x - \mu)) \quad (1)$$

where $\mu = \log(Km'n')/\lambda$, in Eq. (1).

λ and K are the statistical parameters of the ungapped aligned sequence.

m and n are the length of the query sequence and biological database, which can be found using the described formula

$$m' \approx m - \ln(Kmn)/H \quad (2)$$

$$n' \approx n - \ln(Kmn)/H \quad (3)$$

H in Eqs. (2) and (3) represents average expected score/aligned sequences.

For the usual values of $K = 0.13$ and 0.318 , H is considered to take the value 0.40.

We use these usual values of K and H to evaluate the importance of the high segment pair by the technique called a lookup table technique.

Step 10: Combine two or more high segment pair into one larger aligned sequence.

There is a possibility that there exist two or more high segment pair in a database that can be combined to form one single longer aligned sequence. This testifies the relation between the database sequence and the query sequence. In order to contrast the importance of the combined high segment pair region, we have two methods, namely the Poisson method and the sum of the scoring method. Let us consider two sets of combined high segment pair with the respective scores (64, 39) and (50, 45). In order to compare the significance of the two given sets, we can either use the Poisson method or sum of the score. The Poisson method prefers a set of maximum lower score that is 45 is greater than 39 and hence set with (64, 39) is selected. On the other hand, the sum of scores adds the higher score and the lower score of the individual set compares the sum and selects the highest value and hence the first set is preferred.

Step 11: Display the Smith-Waterman gapped local aligned sequence of the query sequence and display the matched biological database sequence.

The first version of the BLAST displays the unmapped aligned sequence with the high segment pair sequence. The advance version BLAST2, however, displays a single aligned sequence along with gaps. These gaps may include high segment pair region.

Step 12: Notify each match whose score is lesser than the threshold score.

4 Role of Machine Learning Technique

4.1 Supervised Learning

The most commonly used approach in machine learning is supervised learning. This technique supervised the system model to make a prediction in accordance with the training data. This technique revolves around the training dataset. The training dataset usually involves sample data which includes the input and the expected outputs that are known already. The data object in the training dataset is referred to as a sample point. These sample points are independent in nature. The training dataset is denoted by $D = \{(X_1, C_1), \dots, (X_n, C_n)\}$ wherein (X_i, C_i) is the sample point of the dataset D . Each sample point is associated with the describing features called as a feature set connote by $\{X_1, \dots, X_d\}$ along with C_i which represent the class of interest. The dataset given to the model is proportioned in a way that represents 70% of the training dataset and 30% as testing dataset. The procedure involved in supervised learning is checked if the model is making a correct prediction or not and improvise as in when the dataset is tested. The correct prediction accounts for the efficiency of the model [1, 5].

4.2 Unsupervised Learning

Unsupervised learning is an approach applicable when the resultant pattern is unknown. It is otherwise called clustering since it partitions the dataset that is fed to it in accordance with their similarities. In nutshell, it is a method of grouping dataset with similar elements together. In contrast with supervised learning, unsupervised learning has no idea about the different classes; rather, it creates classes according to the similarities. The training sample D is connote by $\{X_1, \dots, X_n\}$, and the feature set is connote by $X_i = \{x_1, \dots, x_d\}$ wherein each sample point is associated with the features that describes them completely [1]. The unsupervised learning for bioinformatics can be applied in gene clustering. Clustering of genes from the given

expression samples, we obtain the values of expression and group them in the fashion described above. The target of this approach is to follow natural hierarchy while clustering the dataset [8].

4.3 Feature Selection and Dimension Reduction

The biological data object consists of various features. Each feature represents a dimension. More the features more will be the dimension [9]. In order to perform analysis, the high dimensionality of the biological data must be reduced in a pre-processing step which implies reducing the size of the dataset. There are two main methods that perform this task: (i) dimension reduction and (ii) feature selection.

4.4 Dimension Reduction

This approach entails reducing high-dimensional dataset to a smaller-dimensional dataset that involves most prominent features. This approach solicits principal component analysis that finds the variance of each feature involved in the original biological dataset and selects the feature associated with the highest variance and thereby places it in the first axis of a new space. The selection of the feature associated with the second highest variance, that places it in the second axis, and process continues for the principal component only [3].

4.5 Feature Selection

The objective of feature selection is to specify huge dimension dataset with less number of relevant features. This approach uses a univariate or multivariate filter in the preprocessing step [10]. In these filtering methods, all the features are prioritized to their predefined benchmarks and further their ranks are calculated. The feature with the highest rank is held, and the feature of the lowest rank is rejected. Examples for univariate are Wilcoxon rank sum method, and example for multivariate is Markov blanket filters.



Fig. 8 KDD process [13]

4.6 Machine Learning Approaches: Knowledge Discovery in Database

When we speak about learning, we define learning as acquiring novel knowledge. Discovering information about the things in the existing world, that improves the skill set and performance. Knowledge discovery in the database is the approach that uses machine learning methodology in order to fetch knowledge which is useful from the database [11]. The main conjecture is that the significant knowledge which lies in the database is hidden. Using traditional methods to analyze and interpret the data into knowledge is dependent on the specialist. While this traditional method involving specialist may not be feasible or practical due to the time and price constraints, we rely on machine learning. The principal concept in knowledge discovery is finding beneficial patterns from the database [12]. These patterns are basically rules and information about the properties of the data that occurs in the database. Interpreting and evaluating these patterns steer to the discovery of knowledge (Fig. 8).

4.7 Machine Learning Approach: Decision Tree

Decision tree was formulated by Quinlan. It is otherwise called classification tree or regression tree. It is one of the simplest learning systems that utilize discrete-valued function approximation to judge and classify the datasets. It is used widely because of its simplicity and practicality in machine learning methodology.

The decision tree's learning agent employs divide and conquer technique in order to construct the tree. The dataset along with its properties is fed into the learning agent of the decision tree. The tree then returns either true or false (yes or no) decisions when the dataset is tested. This decision tree is one of the examples for the supervised learning technique. These decision tree approaches are by far easy, practical, and robust; however, overfitting and overlapping are the major drawbacks

of this approach. Similar to a standard tree structure, the decision tree consists of same components: root node, branches, internal nodes, and leaf nodes. The node indicates a test which is performed on it, and leaf node indicates the decision [5].

4.8 Machine Learning Approaches: Genetic Algorithm

The genetic algorithm was invented by Holland. The learning agent in genetic algorithm is indulged on biological evolution theory. The main notion is to uphold the population of the data that stands for the optimal solution to a subjective problem, wherein the solution starts evolving and maturing through competition that is controlled by fluctuation. This improvises the performance parameter of the learning agent [5]. The population is under constant experiment on combination of the two strings or maybe mutation in the strings. The genetic algorithm focuses on making the system adaptive and sustainable in the new environment [14]. And it furthermore finds the best fit among the solution.

The algorithm is subjected to population genetics. The problem to be solved is first considered. The population which is represented as bit strings is formulated randomly from the environment where the learning agent is present. The population is exposed to variation which may indicate that few of the population's individual are a better fit. Evolution takes place iteratively. At each iterative phase, a candidate with better survival proficiencies is selected. This process leads to accepting new candidates for the next iteration and discarding the other which are not selected. The cycle continues until an optimal solution is found. The advantages of genetic algorithm are just like the decision trees which is robust, simple, solves high dimensionality problem, whereas the disadvantage is that it is complex to understand the dynamics in each iteration.

4.9 Machine Learning Approach: Clustering

Clustering is a type of approach that detects similarities in the data and groups them into classes and organizes them. In order to predict and define complex datasets, derivative clustering algorithm is used. There are two types of clustering algorithms: hierarchical clustering algorithm and k -clustering algorithm. In hierarchical clustering algorithm [15], the input dataset is represented in a hierarchical perspective shown in Fig. 9, whereas in k -clustering it collects the input data and allocates each data object to its equivalent group that it belongs to according to the features as depicted in Fig. 10.

There are two approaches followed with a view of clustering the data which is observed. One approach is on the chemical and physical concepts of data's feature set. And the second is on computational and statistical views of the input data. When the dataset is encountered, the agent will cluster them into a group of smaller size

Fig. 9 Hierarchical clustering [13]

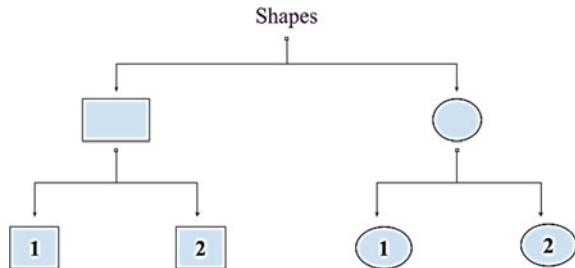
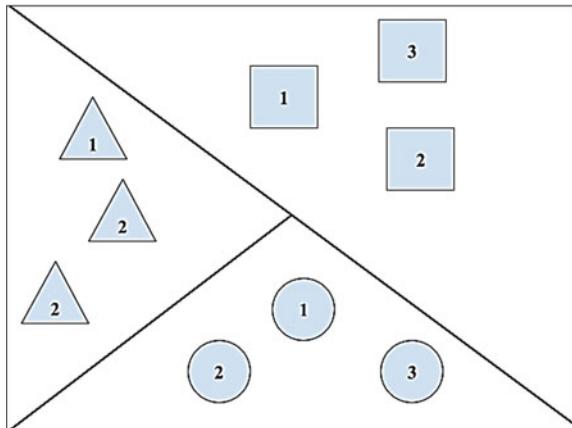


Fig. 10 k -clustering [13]



using either of the approaches. These groups are represented by similar features of the data hence clustering suitable for biological data representation because of its descriptiveness and expressiveness [5].

5 Application of Machine Learning

5.1 Using Clustering Approach to Identify Patients' Subtypes

Some of the serious diseases such as autoimmune disorder, tumors, and cancer are complex to treat since the degree of variation is high in the affected individuals. In that case, precision medicines attempt to resolve this problem by individualizing each of the population and support the creation of medicines of those individuals. It takes gene variation of an individual lifestyle, and the environment of the individual account is considered to predict progression of ailment and stage transition, and it aims to find an appropriate treatment for the individual's ailment.

We use patient subtyping which pays a prominent role in predicting medicines. Here, we distinguish between the population creating subpopulation having similar

patients, and this helps in finding more accurate diagnostic strategies and treatment strategies. This assists not only in medical science but also in practical examination. In reality, from a clinical perspective fine-tuning the prediction process can cut down the uncertainty in the expected upshot of each individual's treatment.

Data integration based on supervised and unsupervised learning was proposed in order to subgroup patients. To advance the accuracy of the agent for better classification of patients, the process tends to use other features such as microRNA (small non-coded RNA found in an organism) expression, and methylation (a process where methyl is added to the DNA) is augmented along with the gene expression. The process of methylation or any copy number alteration acts as a biomarker which is an analogy to bookmark for cancer subtype classification. The data integration effectively helps in subtyping among the population. The main goal is to determine classes of samples which share same or similar yet relevant molecular expressions [6, 13].

An intermediate integration network fusion methodology SNF108 was capable of incorporating microRNA expression and methylation of DNA in order to recognize the subpopulation of the patient. This process takes place in a network structure where at each view patient similarity network is constructed. At each iterative phase, the data arriving from various networks is integrated together in the next upcoming network that yields classes with more similarity at each step. And finally, the iteration seizes to a final fused network.

There are four main steps involved in this approach:

Step 1: Extraction

The data is collected which includes the integrated dataset along with its features that will undergo clustering and dimension reduction to find the most prominent data object of the feature set.

Step 2: Ranking

The prototypes are ranked in accordance with the separability score in order to construct the network.

Step 3: Iteration process

The data from each network is view and clustered according to the similarities at each iteration.

It further optimizes and improves at each coming phase.

Step 4: Network fusion

The final outcome from the various network is clustered which efficiently distinguish between classes accurately by using matrix factorization approach.

5.2 Drug Repositioning Using Classification Approach

The familiar drugs and components may be used to treat new disease other than the disease for which those drugs were targeted to such a process referred to as drug

repositioning. Why drug repositioning is needed? The answer is drug repositioning may reduce failure risk of unfavorable toxicology unlike the traditional method of drug development. The reason we rely on drug repositioning is that the familiar drugs have succeeded in the toxicity test already. Drug repositioning deals with the interaction between the disease and the drug and responses made by cells considering the interaction. Yet these methods experience certain hindrance such as the disorderly structure of the gene expression and other genomic expressions associated with the different ailment [6, 13].

Viewing the biological data in multi-perspective and analyzing the interaction can help the scientific group to experiment with drug repositioning. This process makes use of machine learning and theory of network algorithm in order to examine the data and integrate the information arising from the different networks in iterative phase. The information relates to the similarity of chemical structure, the closeness of the targets in the protein–protein interaction, and correlation of the gene expression after the treatment. In an experiment conducted, a single drug was integrated with gene expression, the structure of the chemical, and drug targets. A kernel-based integration is applied to the resultant view of pictured in a distance matrix. Kernel means are used to create a combined matrix. The first resultant matrix depicts the gene expression correlation, and other two depicts chemical structures and drug target between the protein–protein interactions, respectively. This process results in greater accuracy for the drug repositioning using classification technique.

6 Case Study: Kipoi Utilizing Machine Learning Model for Genomics

Introduction to Kipoi

Kipoi termed as a model zoo for the collection of genomes, in order to promote better usage, sharing, archiving, and building similar models. This model uses deep learning and more extensively machine learning concept in order to comprehend how the sequences of the DNA encode molecular phenotypes which relate to the person's physical appearance. The process of obtaining phenotypes from the gene sequences is called phenotyping. Furthermore, the researchers used machine learning concept to model how the glitch in the code (unmatched sequences) disrupts the prototype which gives rise to the disease. The Kipoi biological database consists of 2000 models from different literature sources and publication.

Making of prediction model

The prediction model mainly includes data-loading and preprocessing. In order to make prediction on new incoming database, Kipoi places four main set points to be followed.

1. Obtaining model parameters,
2. Installation of required software packages,
3. Extraction and preprocessing of appropriate information obtained from various sources,
4. Finally executes the model predictions.

The extraction and the preprocessing of the information in the prediction model and the traditional bioinformatics tool differ from one another with respect to the domains. In genomic [16], the domain-specific dataset is mined as per the required format and the bioinformatics tool help in processing them. The operations like sequencing, extraction of sequences, and annotation (additional description about the sequences) implemented in the bioinformatics tools take days or even weeks to model new data. Hence, the Kipoi aids in rectifying all of these obstacles in order to reduce the time to minutes or seconds.

The main component of the Kipoi

1. Trained model standardization

Trained model standardization includes data loader and data model. The data loader as the name suggests loads the data from standard files with respective formats and preprocesses it which is consumed by the model. It uses python function to load and generate batches of biological data. Specification files included are as follow: dataloader.yaml (for description), dataloader.py (for implementation), and dataloader_files (directory with more required files). Model takes numerous gene sequences and performs predictions. Keras, PyTorch, or TensorFlow are among the frameworks implemented in the model. Specified files included are as follows: model.yaml (for description) and model.py (to implement the classes).

2. Model Repository

These formulated models are situated in the GitHub repository. The parameters and the test files are stored using git large file storage. Hence, the codes are open sourced in order to contribute to the society the same way how GitHub laid its objectives.

3. Accessing model API and using model API

In order to access and use the Kipoi model, an API is shown in Figs. 11 and 12, respectively.

4. Plugins

Plugins provide supplementary functionalities besides executing prediction model and providing access to the model. Hence, we implement plugins to have diversity in prediction and interpretation technique.

API for accessing the kopi models

```
import kipoi

kipoi.list_models() # list available models

model = kipoi.get_model("Basset") # load the model

model = kipoi.get_model( # load the model from a past commit
    "https://github.com/kipoi/models/tree/<commit>/<model>",
    source='github-permalink'
)

# main attributes
model.model # wrapped model (say keras.models.Model)
model.default_dataloader # dataloader
model.info # description, authors, paper link, ...

# main methods
model.dependencies.install() # calls `conda install` and `pip install`
model.predict_on_batch(x) # implemented by all the models regardless of the framework
model.pipeline.predict(dict(fasta_file="hg19.fa",
                            intervals_file="intervals.bed"))
# runs: raw files -[dataloader]-> numpy arrays -[model]-> predictions
```

Fig. 11 Ziga Avsec, API for accessing Kipoi models, GitHub, 2 October 2018

Fig. 12 Ziga Avsec, API for using Kipoi models, GitHub, 2 October 2018

API for using the kopi models

```
# Create and activate a new conda environment
# with all model dependencies installed
kipoi env create <Model>
source activate kipoi-<Model>

# Run model predictions and save the results
# sequentially into an HDF5 file
kipoi predict <Model> --dataloader_args='{"
    "intervals_file": "intervals.bed",
    "fasta_file": "hg38.fa"}' \
-o '<Model>.preds.h5'
```

7 Challenges of Machine Learning Application in Bioinformatics

7.1 Dirty Biological Database

The dirty biological data refers to the redundant data which is an issue that concerns most of the biologist. With the abundant data available over the deposition and exchange over the network to the public domain, it is extremely hard to detect the flaws and maintain the quality of the biological data [13]. This type of dirty data

reduces the accuracy of the agent's programs. The dirty biological data occurs due to certain factors which include:

- Error from the experiment.
- Misleading interpretation.
- Human-made mistake while doing the annotation process.
- No standards followed while interpretation.

Learning about the dirty data and ways to handle it through a machine learning algorithm play out to a plus point in this field of bioinformatics. Furthermore, machine learning approaches help the agent to adapt their learning algorithm to produce an optimal solution or in other words to find the best fit. To tackle with the exponential growth of the biological data and frequently update its short learning and training period is extremely crucial.

In order to overcome these problems, there is a necessity to work along with the specialist in the field like the biologist while the data is analyzed and interpreted. The data in the database has to be updated constantly to maintain the quality of the data. Cleaning the dirty data is one of the solutions that eliminate the non-relevant data from the biological database after analysis. Application of machine learning is one of the challenges that we ought to face in the bioinformatics field.

7.2 *Generative and Discriminative*

Biological data is completely data-driven-based research. The assumption is laid by analysis of the experimental data. The indefiniteness is high when research is based on experimental practice, which make biologist and the data scientist rely more on statistical analysis. We incorporate theories of probability while carrying out bioinformatics analysis that gives a boost to the confidence level. Machine learning approaches yield in either generation of data or discriminating the data by using relevant machine learning technique. Discriminating methods are used to make out the differences between different groups of patterns, whereas general methods are conducted to find intrinsic patterns. However, the selection arises the question. The learning objective and the task that has to be performed conclude which technique should be undertaken. Implementing the correct method will help reduce uncertainty in the given problem statement.

7.3 *Approximation and Explanation*

There is always a discussion when the subject of concern is the issue of approximation and explanation. Some of the approaches of machine learning such as genetic algorithm and neural network generate outcomes without any explanation using a

learning process. This is an allusion to a black box testing, wherein all the testing technique is hidden. Although these technologies result in accurate outcomes, it is often difficult to comprehend strategies and interpret relevant information. Approaches like decision tree may generate an explanation that suits human understandability and also accomplish the task. But there is way too much information at every step. That begs to answer do we need that enormous information to conclude to a final step?

7.4 Single or Multiple Methods

According to the survey conducted, most of the bioinformatics tools utilizes more than one machine learning technique [17–19], since using one technique is vulnerable to break down the entire process. This also helps in attaining significant performance compared to the use of a single machine learning method. Although it provides the performance which is needed, combining these processes is a difficult task since it lacks in coherence. This happens due to the different approaches followed by the output and learning agent.

8 Conclusions

In recent times, the biological research revolves around enormous data which is increasing in a rapid fashion due to high throughput research. The use of a computational tool is crucial in this research which will assist the biologist in order to analyze the sequences of the genome, define the patterns of the sequence, and discover useful information in the biological database. The need for a computational tool gave rise to a new field called bioinformatics along with machine learning. Bioinformatics acts as the interface between the biological research field and computational research.

Bioinformatics in combination with machine learning helps in maintaining the biological database, creates tools that learn to make a prediction, and detects knowledge from the database. The objective of the bioinformatics is to assist in pharmaceutical discovery and discover new ailment and ways to counter it. Machine learning approaches and technique have accelerated bioinformatics research which includes genomic field, proteomic, molecular structure prediction, etc. The use of machine learning concept is inexpensive and efficient to deal with bioinformatics issues. There is a belief that machine learning will play a major role in the bioinformatics future.

References

1. Lai K, Twine N, O'Brien A, Guo Y, Bauer D (2018) Artificial intelligence and machine learning in bioinformatics. <https://doi.org/10.1016/b978-0-12-809633-8.20325-7>
2. Avsec Z, Kreuzhuber R, Israeli J, Xu N, Cheng J, Shrikumar A, Banerjee A, Kim DS, Urban L, Kundaje A, Stegle O, Gagneur J (2018) Kipoi: accelerating the community exchange and reuse of predictive models for genomics. <https://doi.org/10.1101/375345>
3. Gauthier J, Vincent A, Charette S, Derome N (2018) A brief history of bioinformatics. *Brief Bioinform* 2018:1–16. <https://doi.org/10.1093/bib/bby063>
4. Li H (2008) Improvement and application of BP neural network algorithm. Chongqing Normal University
5. Libbrecht MW, Noble WS (2015) Machine learning applications in genetics and genomics. *Nat Rev Genet* 16(6):321–332
6. Tan AC, Gilbert D (2001) Machine learning and its application to bioinformatics: an overview
7. Wang F-L, Song J, Song Y (2009) Application of BP neural network in prediction of protein secondary structure. *Comput Technol Dev* 19:217–219
8. Wong K-C, Li Y, Zhang Z (2016) Unsupervised learning in genome informatics. In: *Unsupervised learning algorithms*. Springer, pp 405–448
9. Hajighorbani M, Reza Hashemi SM, Minaei-Bidgoli B, Safari S (2016) A review of some semi-supervised learning methods. In: *IEEE-2016, first international conference on new research achievements in electrical and computer engineering*
10. Yu X, Yu G, Wang J (2017) Clustering cancer gene expression data by projective clustering ensemble. *PLoS ONE* 12(2):e0171429
11. Satpute BS, Yadav R (2017) Machine learning techniques for bioinformatics and computational biology—a survey
12. Wang YX, Liu K, Theusch E, Rotter JI, Medina MW, Waterman MS, Huang H (2017) Generalized correlation measure using count statistics for gene expression data with ordered samples. *Bioinformatics*
13. Li Q et al (2016) A supervised method using convolutional neural networks for retinal vessel delineation. In: *International congress on image and signal processing*. IEEE, pp 418–422
14. Abdurakhmonov IY (2016) Bioinformatics: basics, development, and future. <https://doi.org/10.5772/63817>
15. Le J (2018) A gentle introduction to neural networks for machine learning
16. Rathore S, Habes M, Iftikhar MA, Shacklett A, Davatzikos C (2017) A review on neuroimaging-based classification studies and associated feature extraction methods for Alzheimer's disease and its prodromal stages. *NeuroImage*
17. Tan AC, Gilbert D (2001) Machine learning and its application to bioinformatics: an overview. August 31
18. Wong KC, Li Y, Zhang Z (2016) Unsupervised learning in genome informatics. In: *Unsupervised learning algorithms*. Springer, pp 405–448
19. Hajighorbani M, Reza Hashemi SM, Minaei-Bidgoli B, Safari S (2016) A review of some semi-supervised learning methods. In: *IEEE-2016, First international conference on new research achievements in electrical and computer engineering*

Text Mining in Bioinformatics



Minal Moharir and Preetham Maiya

1 Introduction

The literatures on various subjects are increasing day by day. And with the help of the Internet, the growth is unprecedented. This is especially true in the field of biomedical sciences. And since the cost of sequencing human genome is under \$1000 [1], biomedical scientists are creating large and complex multidimensional datasets. This has led to an exponential growth in literature. And with this rate of growth of literature raises an issue of efficiently utilizing the data.

The problem of effective usage of data is solved by employing text mining techniques on the available literature. Text mining is a part of data mining, where unstructured data is analyzed and useful information is extracted from them. This enables the literature to be used to promote new discoveries in lesser time and help experts achieve realistic diagnosis with the help of extracted information. Large amounts of data are freely accessible through PubMed, an online search engine to access MEDLINE documents.

There are a lot of approaches to achieve text mining in biomedical fields. Some of them are named entity recognition (NER), document classification and clustering, relationship discovery, hedge cue detection, claim detection, information extraction and information retrieval, and question answering. The main goal of all the techniques is to understand large data on biomedical field in a very short time. To support these techniques, it needs a dataset. The dataset is usually in the form of structured or annotated data like a corpus or a vector of words or word embeddings.

Text mining is the process of extracting useful information from large unstructured data. The data source maybe papers, abstracts, HTML pages, or mails [2]. It uses the process of pattern recognition and lexical analysis for extracting information. Since

M. Moharir (✉) · P. Maiya

Department of Computer Science and Engineering, RV College of Engineering, Bengaluru 560059, India

e-mail: minalmoharir@rvce.edu.in

the data is mainly unstructured, it has to be represented in some sort of structure before applying text mining techniques. It is focused on finding patterns in text and predicting unseen input based on the trained data.

It is applied in various fields like security, online media, sentiment analysis, digital humanities and computational sociology, scientific literature mining, and many more fields. One of the main fields is biomedical field. In biomedical field, it is mainly used for knowledge discovery in texts (KDT), to summarize or annotate unseen biomedical text. Since the terminologies in the medical field are not consistent over many sources, like kidneys maybe replaced with the word renal, ambiguous abbreviations lead to difficulty in implementation of text mining in biomedical field.

Text mining generally contains three steps as shown in Fig. 1: pre-processing, text mining techniques, post-processing [2]. Text is usually unstructured and contains many unwanted words. For cleaning the text and bringing some structure, pre-processing stage is applied. It contains many steps, starting from tokenization, removing stop-words, stemming and converting it to a structured format, usually into high dimensional vector. The main part is the text mining operations used. It can be statistical or may use machine learning models or neural networks. The post-processing step involves selection or interpretation and visualizing the outcome of text mining operations.

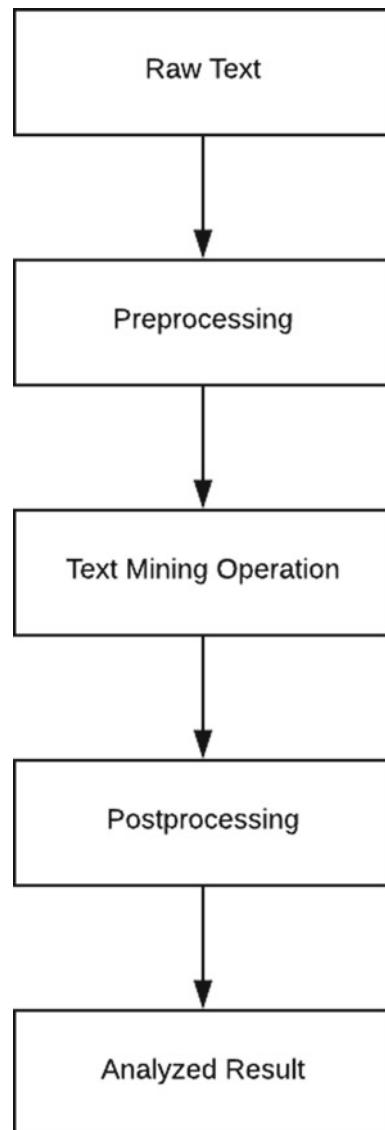
The text mining methods are usually dependent on the domain specified. Each method is developed based on the dataset considered. A truly general text mining model must work on any domain. Due to the difficulty in realizing this goal, different approaches for different domains are taken.

2 Biomedical Application: Case Study

For named entity recognition (NER) of biomedical text, one should take the ambiguity and inconsistency of text into considerations before constructing the model. It also becomes more complex due to long sequences of terms usually given to medical terms. To overcome this, one of the requirements is to have a proper annotated dataset. One of them is The National Center for Biotechnology Information (NCBI). It is a part of United States National Library of Medicine (NLM). To have a good predictive model, dataset is one of the requirements. The other is the architecture used.

The task of named entity recognition (NER) in biomedical field is very different from other fields like news domain or reviews on a subject. Many of the differences give rise to the difficulty in solving this problem in biomedical field. Some of the difficulties faced are:

- The words maybe used differently in the same sentence or across documents.
- They maybe abbreviated in some places or expanded in others.
- Biomedical entities usually span over more than one word.

Fig. 1 Text mining steps

- Many entities sound almost the same or even have similar composition but may belong to different classes.

These are some of the difficulties faced specifically in biomedical domain. We will come across many others with examples as we proceed. Hence, the methods developed are specific to biomedical domain.

There are three general approaches to solving NER: dictionary based, rule based, and machine learning/deep learning. Dictionary based uses a set of manually crafted

dictionary to map the words. But it needs human intervention to update the dictionary and suffers from giving false results due to ambiguity in medical terms. Rule-based approach uses a set of rules for words to be used to classify them. It can give better results compared to dictionary based but the model is very specific to the domain and cannot be ported. Machine learning/deep learning approaches work with wide range of dataset across domains. It is a better approach than the other two.

The approach discussed in paper [3] is entity recognition in electronic health records (EHR). It is a documented finding by the doctors while treating patients. They are usually riddled with incomplete information or typos since doctors do not give them much priority since it will reduce their time with the patients. Thus, this produces a difficulty in performing named entity recognition (NER) by dictionary-based approach.

To overcome the difficulties in achieving this, various different types of dictionary-based annotations are discussed. The first one is the exact match from the dictionary. This would not cover many words due to presence of typos and abbreviations. The second approach is fuzzy matching the dictionary values to the text to be annotated. This takes care of most of the typos in the text. The third approach is by stemmed matching, where stemmed words from text are matched with stemmed words in the dictionaries. This takes care of different forms of the word as the stemmed values are same. All these methods are used for annotating the document. The overall picture is depicted in Fig. 2.

The machine learning approach discussed in paper [4] studies the data set for extracting features from the text. It takes the approach taken in NER in news domain and makes a parallel to the biomedical domain. Although, the entities that are recognized in the domains are different, some of the approaches remain same. The paper lists out the different ways a biomedical entity can be named. They maybe descriptive like ‘normal thymic epithelial cells,’ contain one head noun like ‘91 and 84 kDa proteins,’ which contain two entities, have non-standard names, be abbreviated, or have cascaded entities where one entity lies inside another.

By studying the data, and listing the various forms the entities maybe, the paper chooses various features from each word to be extracted. They are word formation pattern, morphological pattern, POS, head noun trigger, special noun trigger and name alias. It then uses a hidden Markov model for prediction of entities with k-nearest neighbor algorithm for data sparseness. The approach is applied to the GENIA corpus. The overall precision scores achieved on two different GENIA corpus, GENIA V1.1, and GENIA V3.0 is presented in Table 1.

The approach of [5] is direct usage of conditional random field (CRF). It uses the algorithm on two datasets, GENETAG corpora for protein and gene name recognition and BioText corpus for disease name recognition. It uses a type of dictionary-based feature extraction for enhancing the features. Instead of using a dictionary directly, it uses an ULMS metathesaurus to approximate the feature. Direct dictionary extraction is heavily dependent on the domain knowledge. Using the approximate dictionary lookup algorithm, the feature extraction is a bit more flexible. Other features are

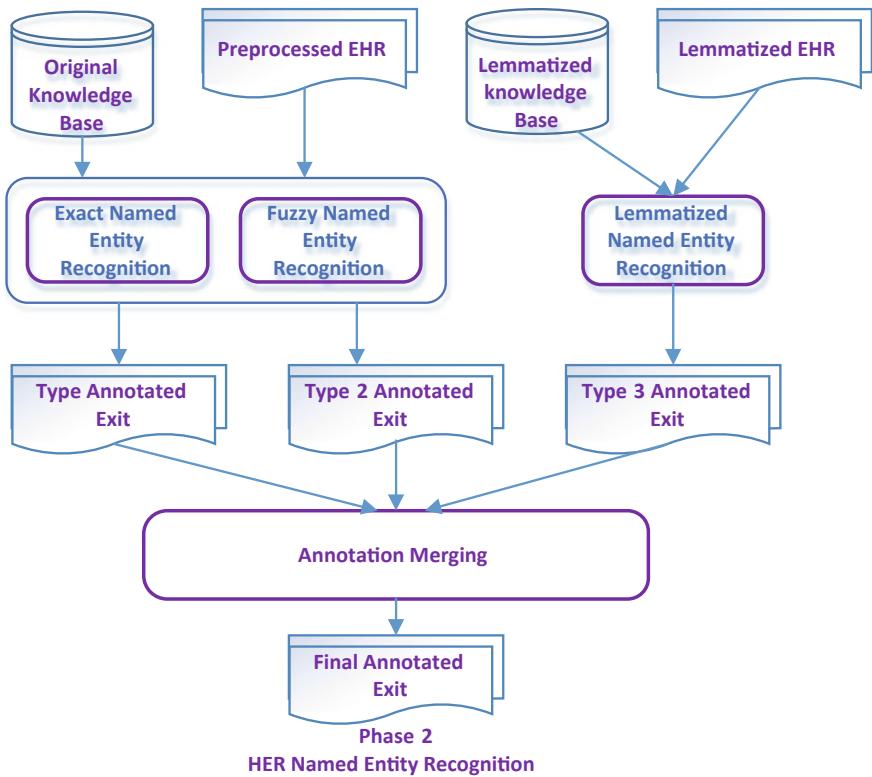


Fig. 2 Combined dictionary based NER

Table 1 Precision scores

Corpus	Precision	Recall	F1
GENIA V3.0	66.5	66.6	66.6
GENIA V1.1	63.1	61.2	62.2

orthographic features like capitalized word, digits or symbols, morphological features like POS, lemma, word shapes, and contextual features. The data set is tokenized and the features are applied to each token.

The high level architecture of the process is shown in Fig. 3. A CRF machine learning algorithm is used for the tokens with the features. This approach uses a 2-order CRF, which depends on the previous two states to predict the next state instead of a normal CRF, which only looks up to one previous state. It provides the evaluation metrics of the two data sets, the disease dataset and the gene dataset.

The three named entity position encoding schemes, namely IO, BIO, and BIOEW are compared using Hepple tagger and MedTagger; the results are shown in Tables 2 and 3. Our results shown in Table 2 suggest that the more complex coding schemes do not necessarily increase the F-score for Biotext corpus NER task. The IO encoding

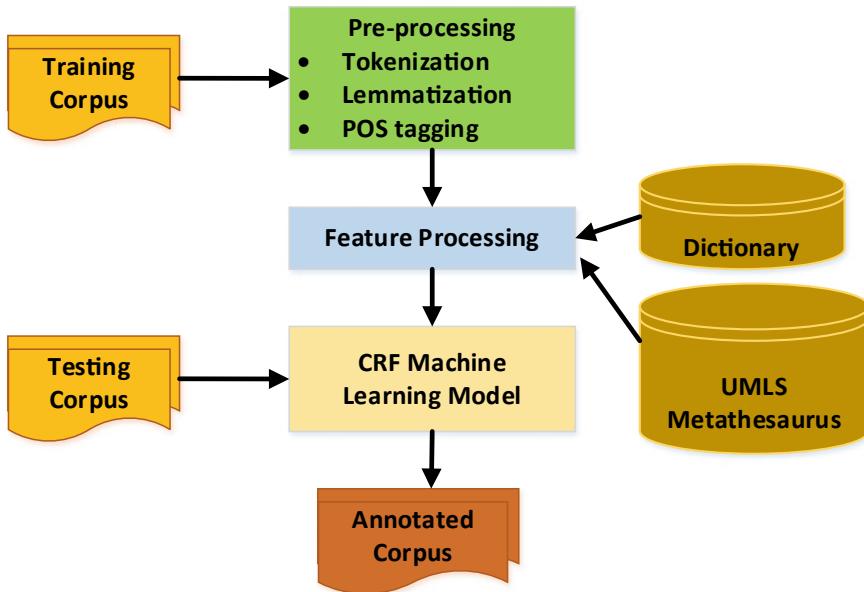


Fig. 3 General architecture of CRF model

Table 2 Results of evaluating different entity encoding scheme on biotext NER task with hepple tagger

Encoding scheme	Precision	Recall	F1
IO	62.82	47.79	54.28
BIO	63.40	47.13	54.07
BIOEW	63.11	46.61	53.61

Table 3 Results of evaluating different entity encoding scheme on biotext NER task with med tagger

Encoding scheme	Precision	Recall	F1
IO	87.42	69.40	77.38
BIO	87.93	82.29	85.02
BIOEW	83.05	74.57	78.58

scheme gives the slightly better F-score than BIO and BIOEW schemes. This is in agreement with the finding in [13] that uses the BioCreative II corpus for gene/protein NER task. In this paper, the IO setting is retained for our experiments.

Further the three schemes were evaluated using MedTagger in Table 3. The F-score is improved using MedTagger. The BIO encoding scheme gives the better F-score than IO and BIOEW schemes using MedTagger.

The approach in [6] is partially based on rule based with machine learning. It defines a set of rules for defining features. It uses support vector machines (SVM) as the machine learning method to classify into different labels. SVMs are supervised

learning algorithm, mainly used for classification. It uses a separating surface to classification. Text classification requires features to be extracted from text to apply on SVM.

It uses parts of speech tags and word shape among other features. Word shape is a set of rules as to how each word should be differentiated based on the composition of the word. For example, if a word contains all upper case letters or contains the symbol ‘\$.’, it is then fed to an SVM model with the proper format. The format followed is as follows:

```
[label][index1] : [value1][index2] : [value2]...
[label][index1] : [value1][index2] : [value2]...
```

The model is trained with the training data and the result is stored in this format for further classification since training is computationally expensive, taking a long time.

The GENIA corpus with annotations was used as the dataset. The results state that for biomedical terms, the precision is 94.33% with recall rate 71.67%. For non-medical terms, the precision score was 69.73% and recall rate is 93.82%. So the overall precision score comes out to be 84.34% and recall score as 80.76%. The main disadvantage with simpler models like this is the scores vary a lot over different datasets used.

The other approach is discussed in paper [7]. It uses a skip-conditional random field (CRF) instead of a standard CRF. It identifies that usually in medical texts, words next to each other may not be belonging to same class. Sometimes, even though they may have similar features, they belong to radically different classes, making the direct use of CRF, give wrong results. Many a times, the related words maybe far apart in the same sentence.

Hence, it employs a skip-CRF model, with having regular connections to nearby words also has extra connections to distant words, which specifically helps in medical texts. The structure of skip-chain CRF is shown in Fig. 4. There are edges going from distant words. For example, hOggl, mOggl, and Oggl do not appear next to each other but are related. The performance of above scheme is tabulated in Table 4.

The approach in [8] adopts various methods with long term short memory (LSTM) network, bi-directional LSTM network, word embeddings, character embeddings, and CRF. It lists out the advantages of adding each of the parts mentioned above. LSTM network help with adding context to current word based on the previous word. Bi-directional LSTM takes words in the future with the words previously looked at, and hence gives a better context. Character embeddings help with the out of vocabulary (OOV) problem. And finally, CRF's help in recognizing the sequence of words which form an entity.

Pretrained word embedding's taken. For character embeddings, two vectors are used: one with reading words forward, v_f and one with reading words backwards, v_b . These vectors are concatenated with the word embeddings and fed to a bi-directional LSTM. Finally, to prevent over fitting, dropout techniques are used. The output of

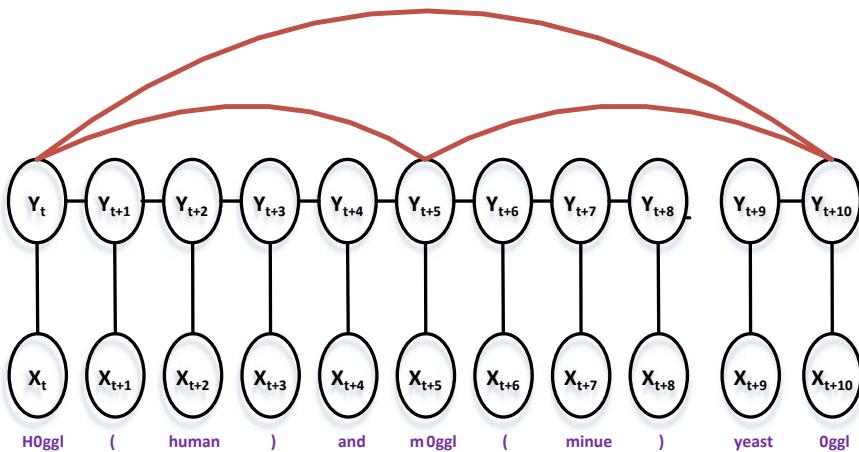


Fig. 4 Skip-chain CRF with linear-chain CRF

Table 4 Results of evaluating skip-conditional random field (CRF)

Precision	Recall	F1
72.8	73.6	73.2

this network is fed to a CRF to extract the entity name. An example of the final network is shown in Fig. 5.

It presents the scores of various architectures in Table 5. They are as follows.

- LSTM Long short-term memory
- WE Word embeddings
- BLSTM Bi-directional long short-term memory
- CE Character embeddings
- CRF Conditional Random field

The paper [9] introduces a novel method, crafted specifically for biomedical data and has an implementation based on Python programming language. It uses TensorFlow for building neural network and word2vec model for representing text in a vector. It [9] uses a novel approach by using a convolutional neural network (CNN) with POS tagging and word and character embeddings to create a new type of network which it has named as ‘GRAM-CNN’ as shown in Fig. 6. It hypothesizes that long short term memory (LSTM) network, based on recurrent neural network (RNN), which is usually used for tagging, would not give high performance since long sentences would often contain information unrelated to the current sentence and would degrade the efficiency of LSTM. This approach concatenates POS tag, word embeddings and character embeddings with the word itself and feeds this to a CNN. Word embeddings alone cannot give useful representation for out of vocabulary words. To help with this, character embeddings are used. It was implemented using a CNN.

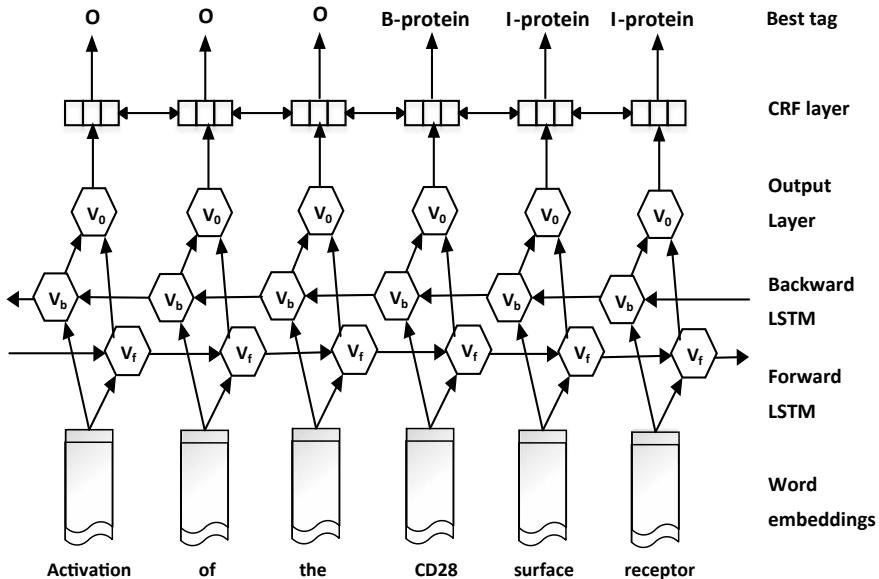


Fig. 5 Bi-directional LSTM with character and word embeddings and CRF

Table 5 Results of evaluating different architectures

Models	Precision	Recall	F1
LSTM + WE	68.99	72.72	70.81
BLSTM + WE + CE	72.78	74.22	73.49
BLSTM + WE + CE + dropout	75.22	73.58	74.39
BLSTM + WE + CE + dropout + CRF	74.16	77.66	75.87

The general architecture of the CNN is shown in Fig. 7. The output of CNNs is then feeded to a conditional random field (CRF) to give a final output. Since, biomedical entities contain more than one word, CRF is used to predict a sequence of words as an entity. To predict the sequence label, sequence having the largest probability was chosen. There are also different computational methods [10–12] which have been developed for protein structure prediction or essential protein prediction using machine learning methods. The prediction scores on NCBI dataset are as follows:

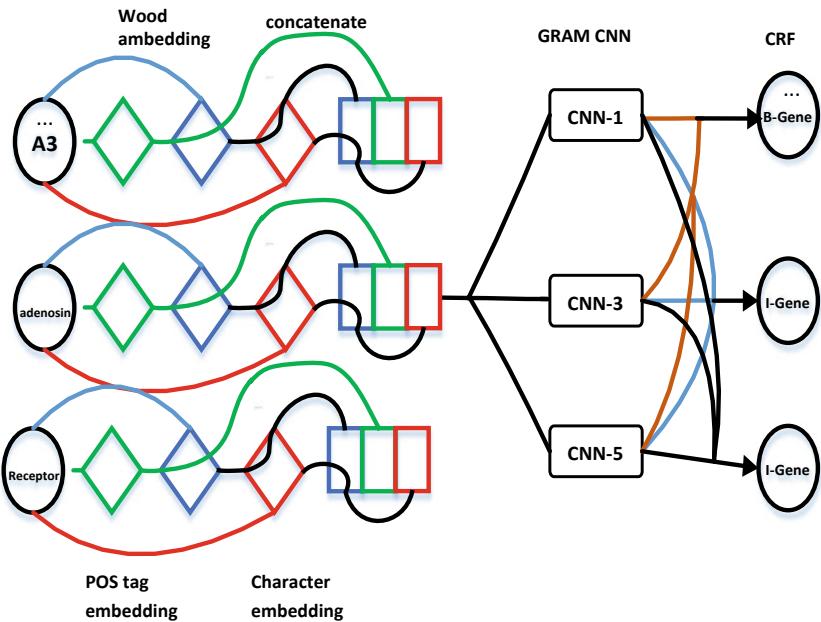


Fig. 6 Architecture of GRAM-CNN

3 Conclusion

This chapter has discussed what text mining is and how it is important for biomedical applications. Further it provided a comprehensive overview of text mining methods. The chapter discussed various methods with different case studies. The results from different case studies are analyzed in detail.

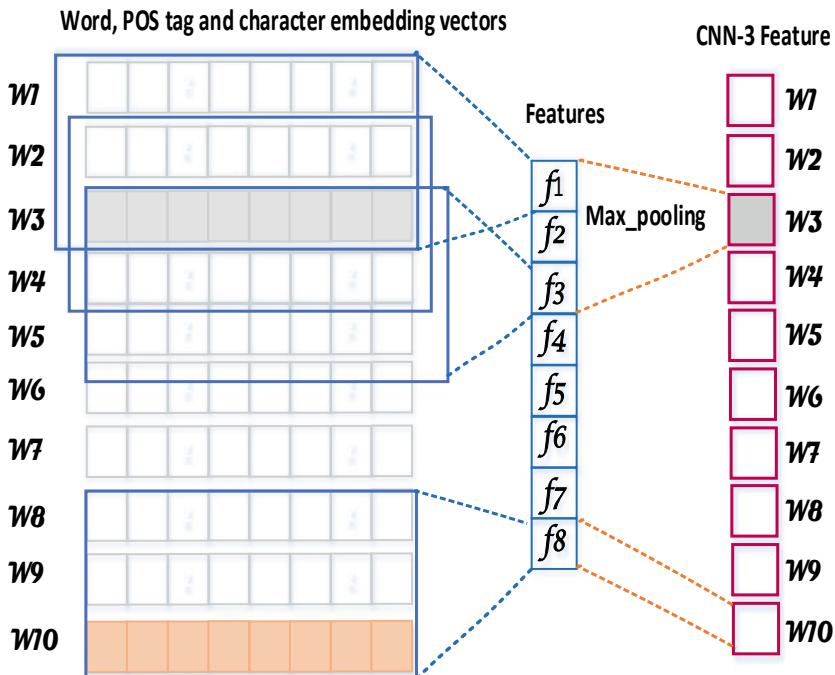


Fig. 7 Architecture of CNN

References

- Goldfeder RL, Wall DP, Khoury MJ, Ioannidis JPA, Ashley EA (2017) Human genome sequencing at the population scale: a primer on high-throughput DNA sequencing and analysis. *Am J Epidemiol* 186(8):1000–1009
- Zhang Y, Chen M, Liu L (2015) A review on text mining. In: 2015 6th IEEE international conference on software engineering and service science (ICSESS), Beijing, pp 681–685
- Quimbaya AP et al (2016) Named entity recognition over electronic health records through a combined dictionary-based approach. *Procedia Comput Sci* 100:55–61
- Zhou G, Zhang J, Su J, Shen D, Tan C (2004) Recognizing names in biomedical texts: a machine learning approach. *Bioinformatics* 20(7):1178–1190
- Kanimozhi U, Manjula D (2017) A CRF based machine learning approach for biomedical named entity recognition. In: 2017 Second international conference on recent trends and challenges in computational models (ICRTCCM), Tindivanam, pp 335–342
- Ju Z, Wang J, Zhu F (2011) Named entity recognition from biomedical text using SVM. In: 2011 5th International conference on bioinformatics and biomedical engineering, Wuhan, pp 1–4
- Liao Z, Wu H (2012) Biomedical named entity recognition based on skip-chain CRFS. In: 2012 International conference on industrial control and electronics engineering, Xi'an, pp 1495–1498
- Gridach M (2017) Character-level neural network for biomedical named entity recognition. *J Biomed Inf* 70:85–91
- Zhu Q, Li X, Conesa A, Pereira C (2017) GRAM-CNN: a deep learning approach with local context for named entity recognition in biomedical text. *Bioinformatics* 34(9):1547–1554

10. Sekhar SM, Siddesh GM, Manvi SS, Srinivasa KG (2019) Optimized focused web crawler with natural language processing based relevance measure in bioinformatics web sources. *Cybern Inf Technol* 19(2):146–158
11. Sekhar M, Sivagnanam R, Matt SG, Manvi SS, Gopalalyengar SK (2019) Identification of essential proteins in yeast using mean weighted average and recursive feature elimination. *Recent Patents Comput Sci* 12(1):5–10
12. Patil SB, Sekhar SM, Siddesh GM, Manvi SS (2017). A method for predicting essential proteins using gene expression data. In: 2017 International conference on smart technologies for smart nation (SmartTechCon), pp 1278–1281. IEEE
13. Sukanya M, Biruntha S (2012) Techniques on text mining. In: 2012 IEEE international conference on advanced communication control and computing technologies (ICACCCT), Ramanathapuram, pp 269–271

Open-Source Software Tools for Bioinformatics



T. Gururaj and A. P. Pavithra

1 Introduction

Recently, Information and Communications Technology (ICT) were highly used in the domain of cloud deployments, social media, mobile applications, web servers and data transmission. IT has been involved in the various fields of the corporate and social world including medical sciences [1]. Most advanced equipments were available in the medical laboratories for accurately diagnosis and analysis of the human body and diseases. These diagnosis machines involve in the electroencephalography (EEG), computed tomography (CT), magnetic resonance imaging (MRI), etc. These devices are provided more information about the human body and assisted the medical experts for the diagnosis and selecting the suitable course of treatment.

With diagnostic machine, software tools and libraries are also used. Biological data collected by the computer diagnostics machines were analysed by the software tools. The concepts of bioinformatics have evolved that involve in the software tools and application for the analysing the medical and biological data for the diagnosis. These softwares use the powerful and high-end algorithms in the backend process for analysing the large collection of the medical data and that lead to better medical treatments. The molecular biology field is closely related to the bioinformatics, and the process was in similar to the biological structure analysis and diagnosis. Molecular biology involves in the process of deep analysis of the body cell movement with related to the proteins, biosynthesis, RNA and DNA.

T. Gururaj (✉)

Research Scholar, Department of ISE, MSRIT, Visvesvaraya Technological University, Jnana Sangama, Belagavi 560054, India

A. P. Pavithra

Associate Professor, Department of ECE, SJM Institute of Technology, Chitradurga 577502, India

T. Gururaj · A. P. Pavithra

Visvesvaraya Technological University, Belagavi, Karnataka, India

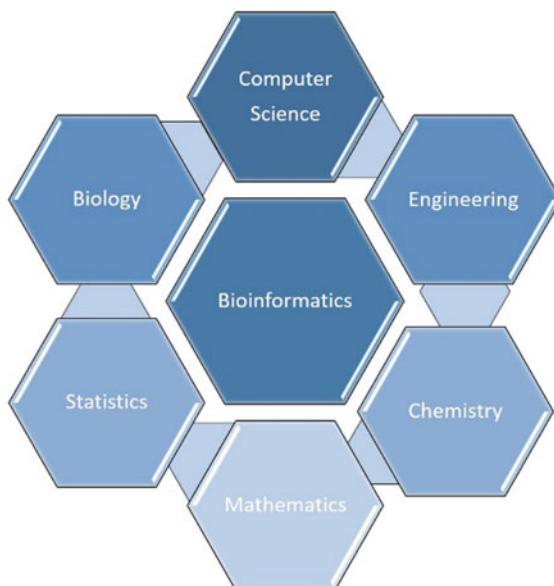
Many medical data sets were available for the research, which are provided from the diagnostic laboratories. These data sets can be used by the researchers to study the architecture and data structure of biological data. The programmers can able to download these medical data from online and apply the effective algorithm for analysing the hidden information in the data for the diagnosis.

Recently, bioinformatics and predictive analysis in the medical field have gained more attention for the related applications. The signals extracted from the human brain and heart are applied for the information extraction, which is used for processing and predictive analysis. The data set from the FPMS, UCSD, PhysioNet and others can be applied for the bioinformatics investigation based on the machine learning tools.

Figure 1 illustrates that bioinformatics is the interdisciplinary subject including chemistry, engineering, biology, computer science and statistics for analytics and predictions. Molecular biology is helpful in studying biological structures.

With computer tools and algorithms, the medical and diagnostic sciences were analysed by the computer professionals. Even researchers are interested in interdisciplinary field of bioinformatics so that their programming skills can be used for the medical sciences.

Fig. 1 Elements of bioinformatics



1.1 Open-Source Software Tools

Medical data were analysed using the software tools to analyse its structure. Some of the tools based on specific datasets were listed below [2].

OpenEEG (<http://openeeg.sourceforge.net/doc/>): OpenEEG is the open-source online tools that can be used to analyse the EEG signals. This tools MIR and Brain-Wave Viewer. EEGMIR (<https://sites.google.com/site/eegnetworks/>) is the open-source tool for analysing the EEG brain signals. The brain network can be viewed with these features.

BioSig (<http://biosig.sourceforge.net/>): BioSig is open-source which consists of many libraries as add-ons such as Brainathlon, NeuroServer, BrainBay, BioEra, EEGce software libraries with many features to process the signal. The library has useful features in analysing the signal including EEG, electrooculogram (EOG), electrocorticogram (ECoG), electrocardiogram (ECG), electromyogram (EMG), respiration and others. Additionally, this can support the toolbox interface to the popular softwares like MATLAB, Python, Octave, PHP, Perl, C, C++, Tcl and Ruby. The medical domains such as neurophysiology, cardiovascular systems, neuroinformatics, psychology and sleep research require brain-computer interface, which are effectively processed by BioSig.

GenomeTools (<http://genometools.org/>): GenomeTools is the open-source tool for the analysis of the genome and biological parameter analysis. The libraries are available for the bioinformatics and can support C language with APIs with detailed instruction. The GenomeTools can be used for the deep analysis of the biological structures.

Biopython for molecular biology (<http://biopython.org/>): Biopython provides the set of tools and libraries for computation of the biological structures. The open-source distribution is promoted by the Open Bioinformatics Foundation (OBF). Python code is used to convert the bioinformatics files into data structures for analysis. International formats that are supported in Biopython: UniGene, PubMed, GenBank, Medline, GenBank, FASTA, ClustalW and BLAST. BioSQL (<http://biosql.org/>) can be used with Biopython to store a biological database.

Bioperl: Bioperl is the open-source software with free of cost and is licensed under the Perl Artistic Licence (<http://www.perl.com/pub/a/language/misc/Artistic.html>). The software can be downloaded in the link of <http://www.bioperl.org>.

The Bioperl object model is tested to support the enterprise-level application such as Ensembl that provides easy learning curve for Perl programmers. This software is able to analyse and process the results from software such as ClustalW, BLAST or the EMBOSS suite. BioCORBA can be used in this software to support the modules written in the Java and Python language, which supplementary materials are available at www.genome.org.

1.2 *Interoperability*

In some cases, the bioinformatics problems were solved by the best solution of hybrid tools. Supporting interoperability [3] between various languages enables a programmer to use different toolkits to develop the component based on the work done by other languages and projects. These are written in various programs like Java, Python and C and can support Perl program by invoking, which is often referred to as shelling out.

Bioinformatics community accepted data structures and number of Extensible Markup Language (XML) formats are supported by the Bioperl.

Software interoperability is not only processed using the external programs, but also uses the remote components possibility developed in different programming language from calling component. Bioperl is compliant with the BioCORBA (Common Object Request Broker Architecture) project (<http://www.biocorba.org>) and is one of the standard techniques of CORBA components for biological process. BioJava and Biopython also support the BioCORBA project.

1.3 *Summary*

Biomedical and bioinformatics are two domains in research for predictive analysis and various applications innovated. Data mining with machine learning techniques on signals generated by different parts of human body can be evaluated. With this introduction, we will consider several tools in detail in the coming topics.

2 **Bioperl**

Bioperl is open-source which has gained international standards for biologists [4], scientists and bioinformaticians that have evolved since from past so many years and approached with a complete library of Perl modules for the usefulness of life sciences. Bioperl offers easy, constant and reliable programming for bioinformaticians. Complexity of the program can be reduced with simple line of code by using Bioperl.

2.1 *Introduction to Bioperl*

Bioperl toolkit provides reusable Perl modules which contain generalized routines related to the life science information. The main objectives of the writing the toolkit

are to focus on a solution that shares the components instead of duplication effort [5].

Codes are freely available in the open-source licence, so others could extend the routines in the Bioperl library and contribute their routines as well. Just as Human Genome Project shares the public data sharing, the open source of the Bioperl project, which minimizes the time for the solutions and new tool to reach the community.

The aim of Bioperl is to allow the user to focus on the problem, such as the logic needed to filter hits in a BLAST [6] represents by certain aspects, instead of actual mechanics in the parsing BLAST report.

The Perl modules are shown in Table 1 and are organized by logical names. For instance, the Bio_Search can be used to search the database, and Bio_Graphics has the modules related to the drawing. The Bioperl modules are the simplified API, and this provides the common Bioperl functions.

Most advanced use of the Bioperl is depended on the Ensembl project [7]. The sequence handling, sequence features and file format parsing are developed for the automatically annotating the various genomes. The Bioperl sequence, feature pair objects and sequence features can be implemented using this Bioperl.

Table 1 Bioperl modules [3]

Modules	Purpose
Seq	Sequence properties
SeqIO	Sequence I/O
Index	Indexing and retrieval
DB	Accessing database using HTTP
DB_GFF	SQL GFF database for DAS and GBrowse backends
SeqFeature	Feature extraction based on sequence location
Annotation	References and comments
AlignIO, SimpleAlign	Sequence alignments and their I/O
LiveSeq, Variation	Variations in sequences
Search, SearchIO	Database searches and their I/O
Tools	Miscellaneous
Tools_Run	Wrapper for executing local and remote analyses
Tree, TreeIO	Phylogenetic trees and their I/O
Structure	Protein structure
Map, MapIO	Biological maps and their I/O
Biblio, DB_Biblio	Bibliographic References and Database retrieval
Graphics	Displays sequences in graphical format

2.2 Why Bioperl?

Many problems are in the computation biology such as file parsing and string. Solving these problems is a major task among the developers in bioinformatics. Since there is no standard method for processing the biological data, different developers follow their own program and own file format to compute that creates difficulties. For instance, this is difficult to process the same data into different laboratories. Although most of the laboratory equipments are similar and try to solve the same problem, this is difficult to process the same data in the computation experiments due to the different file format and softwares.

Bioperl projects aim to co-ordinate the effects of the bioinformatics developers to build the standard tool to process the genomic analysis. Bioperl consists of the modules and documentation in online. Each module consists of one or more object. The main objects of Bioperl are:

- Sequence
- Sequence Alignment
- BLAST
- Alignment Factories
- 3D Structure.

2.3 What Can You Do with Bioperl?

- Sequence data can be accessed from local and remote area
- Transform the database formats/file records
- Manipulate individual sequences
- Search similar sequences
- Creating and manipulating sequence alignments
- Gene and other structure search on genomic DNA
- Developing machine readable sequence annotations.

2.4 Bioperl Installation

Bioperl's large module collections can be easily installed. The additional steps are also provided for the outside program and Perl depends on Bioperl. Probably, Perl's CPAN is needed to fetch and install Bioperl.

“INSTALL” is a document that provides the step-by-step instruction to install Bioperl on the various operating systems of Window, Linux and Mac. It is the part of Bioperl distribution and can be found at <http://bioperl.org/Core/Latest/INSTALL>.

This location may change, but it is easy to find the document in the Bioperl download page.

Before installing the Bioperl, reading the document first was recommended. Here, provide an overview of the installation, and comments among common installation are given below:

Installing Bioperl for the different platforms is given below:

For Unix/Linux: You need to download tar file from the website and untar it, go through configure. Follow the step-by-step procedures that are provided in the “INSTALL” document, which present in the distribution.

For Microsoft Windows: In ActiveState’s Perl (<http://www.activestate.com>), there is a PPM file present in Bioperl; current present at <http://bioperl.org/ftp/DIST/Bioperl-1.2.1.ppd>.

Bioperl consists of CVS repository that fetches the current version of the modules. Some newer version consists of the new features, and more bugs can be found, the less bugs in the older and stable version releases. The details to install CVS repository are provided in the Bioperl website.

All these methods to install Bioperl are ease and most general way for Perl programmers to install module set based on CPAN.

Bioperl can be installed by providing the following command line:

```
perl -MCPAN -e shell;
```

This opens the CPAN shell prompt:

```
cpan>
```

Module is needed to install often which requires other modules for its proper functions, and one or more required modules may not be installed. The CPAN enables way to check whether it is other required modules which are installed, and the steps can be followed to install the missing prerequisites.

In Bioperl for large collection of modules, prerequisites may pop up. This can be configured by the following command:

```
cpan> o conf
```

The query from the output lines was:

```
prerequisites_policy ask
```

List of options in the CPAN sessions was not found and CPAN documentation is analysed and checked for the string prerequisites_policy, and read the following:

```
prerequisites_policy
what to do if you are missing module prerequisites
('follow' automatically, 'ask' me, or 'ignore')
```

To automatically handle prerequisites, CPAN can be configured as:

```
cpan> o conf prerequisites_policy follow
```

Internet connection may get slow as many prerequisite may have to be set. Incase if the system gets hung, wait till the installation of program completes as it takes longer time.

CPAN session command can be summarized by the following commands:

```
cpan> help
Display Information
command argument      description
a,b,d,m WORD or /REGEXP/ about authors, bundles, distributions, modules
i      WORD or /REGEXP/ about anything of above
r      NONE            reinstall recommendations
ls     AUTHOR          about files in the author's directory
Download, Test, Make, Install...
get              download
make             make (implies get)
test    MODULES,       make test (implies make)
install   DISTs, BUNDLES make install (implies test)
clean             make clean
look              open subshell in these dists' directories
readme            display these dists' README files
Other
h,?      display this menu      ! perl-code  eval a perl command
o conf [opt]  set and query options q      quit the cpan shell
reload cpan   load CPAN.pm again      reload index  load newer indices
autobundle   Snapshot           force cmd    unconditionally do cmd
cpan>
```

The Bundle::BioPerl has some extra useful modules in the Bioperl uses. Bundle is first installed for the further process:

```
cpan> install Bundle::BioPerl
```

This will take the module code from the repository, then unpack, process and install the module with the prerequisites. The main Bioperl distribution is installed.

Latest release of the CPAN is in 1.2.1 version. To analyse the latest release of the module, websites were read the news about the latest releases.

Analyse the INSTALL file to check for the process. The Perl version 5.8.0 is one of standard supported platforms.

Installation is tested on the notebook computer consist of Intel 686 process with the operating system of Red Hat Linux 7.2. The operating system and computer of two years old were used to test the installation and Bioperl website was checked and this recommends the versions of Linux.

Generally, the modern computer system is complex and they are changing rapidly. The hardware and operating systems have the replacement cycle of about two years. The system is needed to be in sync to perform the function. The Bioperl and Perl are needed to co-ordinate with the hardware and software; other modules like web server, C compiler, web browser and others may cause some problems. The system requirement is needed to be analysed for the installation process.

However, no warning is provided in this installation environment. The latest version of Perl is installed with success due to diligence installation. Then, Bioperl installation is performed.

Type the command for the installation:

```
cpan> install B/BI/BIRNEY/bioperl-1.2.1.tar.gz
```

CPAN follows a great amount of activity from the distribution of the Internet and analyses the various modules. The modules were tested, a few of them failed and CPAN does not install the modules. After the few failures of the installation process, check the failures in the peripheral components of Bioperl and this can be fixed them later.

To complete the installation, CPAN is forced to install despite some failures in the test process:

```
cpan> force install B/BI/BIRNEY/bioperl-1.2.1.tar.gz
```

This resulted in the modules and the documentation being installed.

2.5 Sequence Object

Bioperl consists of the number of sequence object, and most common is Seq. This denotes the single nucleotide sequence and processed in the file of Seq.pm. This

is widely used in Bioperl object due to every program that generates, modifies and processes the DNA sequences using sequence object. This is automatically created when you read a sequence from the file or a database. The object has the techniques for the writing and reading the data into the different file formats. The file formats such as Raw, GenBank and FASTA are supported. The basic operations such as sequence translation, DNA to RNA conversion and subsequence extraction are supported in the sequence object.

Example of creating a sequence object is shown below:

```
$seq = Bio::Seq->new('-seq'=>'actgtggcgtaact',
                      '-desc'=>'Sample Bio::Seq object',
                      '-display_id' => 'something',
                      '-accession_number' => 'accnum',
                      '-alphabet' => 'dna' );
```

The sequence is read from the file. The following function converts the sequence file from the FASTA format into text raw file format [8]:

```
$in = Bio::SeqIO->new('-file' => "inputfilename",' -format' => 'Fasta');

$out = Bio::SeqIO->new('-file' => ">outputfilename",' -format' => 'raw');

while ( my $seq = $in->next_seq() ) {$out->write_seq($seq); }
```

Some other useful sequence objects are as follows:

```
$seqobj->revcom # reverse complements sequence
$seqobj->translate # translation of the sequence
```

The last function processes the same function as the last Perl program. This translates DNA strings into amino acid string.

3 Biopython

Biopython is one of the most common bioinformatics packages of the Python. Various sub-modules were present for the common bioinformatics process [9]. This is developed by Chapman and Chang, mainly developed in Python. This consists of the C code to optimize the complex software process. This is compactable for the Linux, Windows, Mac, etc.

3.1 Introduction

Biopython consists of various Python module, mainly developed to process the data related to the DNA, RNA and protein sequence operations such as identify the modified protein sequence and reverse complementing of a DNA string. A lot of parse is provided to process the common databases like GenBank, FASTA, Swissport, etc. and also provide the interface to the execute the other popular bioinformatics tools like NCBI, Entrez, etc. in the Python environment. This has sibling projects like BioJava, Bioperl and BioRuby.

The Biopython is the International Developer Association project freely available in the Python (<http://www.python.org>) for computing the molecular biology process. The website <http://www.biopython.org> provides the online resources for the Python-based software developer in bioinformatics to process the scripts, modules and weblinks. Basically, this tool is developed to program in Python and helps to develop the high-quality reusable modules and scripts [10].

The main Biopython release supports lots of functionality, including:

- Parse bioinformatics file in Python using data structures
- Supported format files were iterated over record by record
- Develop the code to process online bioinformatics data
- Provide interface for the common bioinformatics programs
- Establish common standard sequence class
- Tools for performing common operations
- Connect with BioSQL, which is a sequence data set that supports the Bioperl and BioJava projects.

3.2 Goals

The Biopython is developed to provide simple, standard and extensive access in the bioinformatics in the Python language. The Biopython-specific goals were listed below:

- Providing standardized access to bioinformatics resources.
- Establish high-quality, reusable modules and scripts.
- Fast array manipulation based on cluster or classification models.
- Genomic data analysis.

3.3 Advantages

Biopython requires less code and also has the following advantages:

- Provide microarray data type used in clustering

- Support modification in Tree-View type files
- Support structured data applied for PDB parsing, analysis and representation
- Support Medline application journal data
- Support standard BioSQL database
- Provide modules to support parse development for bioinformatics file in a specific format
- Cookbook style is used to clear format.

3.4 Biopython Installation

The installation process is easy and typically, and this requires five minutes.

Step 1—Verifying Python Installation

Biopython is developed to support the Python 2.5 or higher version. Basically, Python is to be installed on your system. Execute the following command to check for Python version:

```
> python --version
```

It is defined below

```
C:\Users>python -version  
Python 3.6.5
```

This shows which version of the Python is installed. If the older version is present, download the latest version, install it and run the command again.

Step 2—Installing Biopython using pip

Biopython can be easily installed based on pip by the command line for all platforms. Execute the command below:

```
> pip install biopython
```

The following response is shown in the screen

```
Collecting biopython
  Using cached
  https://files.pythonhosted.org/packages/6a/22/c5b6edsf3j3jlgjfik4asl
  hfklssue6klahf/biopython-1.72-cp........................
    Requirement already satisfied: numpy in
    /Library/Frameworks/Python.framework/versions/3.6/site-packages (from
    biopython)(1.14.2)
    Installing collected packages: biopython
      Successfully installed biopython-1.72
```

Older version of Biopython is updated by

```
> pip install biopython --upgrade
```

The following response will be shown on your screen:

```
C:\Users>pip install biopython -upgrade
Requirement already up-to-date: biopython in c:\program
files\python36\lib\site-packages (from biopython)
C:\Users>
```

Once the command is processed, the older versions of Biopython and NumPy are removed, and recent versions were installed.

Step 3—Verifying Biopython Installation

Biopython is successfully installed on the system. To analyse Biopython properly installed, the process of the command on the Python consoles:

```
C:\Users\User\python
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2019, 17:00:18)[MSC
V.1900 64 bit(AMD64)]on
win32 Type "help", "copyright","credits" or "license" for more
information.
>>>import Bio
>>>print(Bio.__version__)
1.72
>>>
```

This shows which version of Biopython is installed

Alternate Way—Installing Biopython using Source

To install Biopython using source code, follow the below instructions:

Download the recent release of Biopython from the following link: <https://biopython.org/wiki/Download>

As of now, the latest version is Biopython-1.72.

Download and uncompress the file and move to the source code folder. Type the below command:

```
> python setup.py build
```

This will build the Biopython from the source code, as below:

```
D:\biopython-1.72>python setup.py build
Running build
Running build_py
Creating build\lib.win-amd64-3.6\Bio\codonalign
Copying Bio\codonalign\chisq.py      ->      build\lib.win-amd64-
3.6\Bio\codonalign
Copying Bio\codonalign\codonalignment.py  ->  build\lib.win-amd64-
3/6\Bio\codonalign
Copying Bio\codonalign\codonalignalphabet.py ->  build\lib.win-amd64-
3/6\Bio\codonalign
Copying Bio\codonalign\codonseq.py       ->  build\lib.win-amd64-
3/6\Bio\codonalign
Copying Bio\codonalign\__init__.py      ->  build\lib.win-amd64-
3/6\Bio\codonalign
Creating build\lib.win-amd64-3.6\Bio\Compass
Copying Bio\compass\__init__.py        ->  build\lib.win-amd64-
3/6\Bio\Compass
Creating build\lib.win-amd64-3.6\Bio\Crystal
Copying Bio\Crystal\__init__.py        ->  build\lib.win-amd64-
3/6\Bio\Crystal
Creating build\lib.win-amd64-3.6\Bio\Data
```

Test the code with the following command:

```
> python setup.py test
```

```
D:\biopython-1.72>python setup.py test
Running test
Python version: 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2019, 17:00:18)[MSC
V.1900 64 bit(AMD64)]
Operating system: nt win32
Test_Ace . . . ok
Test_Affy . . . ok
Test_AlignIO . . . ok
Test_AlignIO_ClustalIO . . . ok
Test_AlignIO_EmbossIO . . . ok
Test_AlignIO_FastaIO . . . ok
Test_AlignIO_MauveIO . . . ok
Test_AlignIO_convert . . . ok
```

Finally, install using the below command:

```
> python setup.py install
```

```
D:\biopython-1.72>python setup.py install
Running install
Running build
Running build_py
Running build_ext
Running install_lib
```

3.5 Sample Case Study

Some of the use cases are to test and analyse the importance of the Biopython in the field for the process of RNA structure, population genetics, etc.

Population Genetics

Population genetics is the study of the genetics variation within population and investigates in the changes in the genes frequencies and allies in the population over spatial and temporal manner.

In Biopython, Bio.PopGen module is for the population genetics. This module contains all the necessary functions to gather the information about conventional population genetics process.

RNA Structure

The three major biological macromolecules are essential for our life which are DNA, RNA and Protein. Proteins are considered as workhorses of the cell and play an important role as enzymes. DNA is a blueprint of the cell and contains all genetic information that requires for cell growth, nutrients intake and propagate. RNA acts as “DNA photocopy” in the cell.

The module of Bio.Sequence objects in the Biopython is used to represent nucleotides, which can be used to develop the blocks of DNA and RNA.

3.6 Sample Code for Sequencing

Biopython is developed to assist the programmer by providing reusable libraries, and the programmer can focus on analysing the sequence instead of developing. The central object in the bioinformatics is the sequence.

Most of the time sequence means a continuous string of letters like “AGTACACTGGT”. Such sequence can be created using the Seq object, by the following commands:

```
>>> from Bio.Seq import Seq
>>> my_seq = Seq("AGTACACTGGT")
>>> my_seq.Seq('AGTACACTGGT', Alphabet())
>>> print my_seq AGTACACTGGT
>>> my_seq.alphabet Alphabet()
```

Common or sometimes first step in analysing the data in bioinformatics is to extract information from the biological databases. Accessing the databases manually requires more time and complicated process, especially if one needs to analyse the many information. Biopython provides the online database access from the Python scripts, which helps to save time and energy. Currently, Biopython supports the following databases to extract required information:

- Entrez (and PubMed) from the NCBI.
- ExPASy.
- SCOP—See the Bio.SCOP.search() function.

4 Conclusion

Larger development in the new bioinformatics tools for infrastructure is witnessed, due to the machine learning and analysis algorithms. Productive development of new tool, with user interface and documentation, helps the scientists. Most of the available tools are open source, which enables them to process other applications and pipelines with reusable modules. The programmer can primarily focus on genomics toolkits. The development of open and freely available tools helps the scientists in bioinformatics to integrating data analyses. Future work will involve in developing the infrastructure and provide more robust analysis of tools for bioinformatics.

The gentle introduction about the Bioperl provides a framework for the biologist and analysing the scripting in general. Generally, large amount of data are usually generated in research, such as sequence and expression data. The Biopython might become basic tool for the research in general and not only bioinformaticians. From the analysis, we believe Bioperl and Biopython are good stepping stone for bioinformatics.

References

1. Stajich JE (2006) Open source tools and toolkits for bioinformatics: significance, and where are we? *Brief Bioinform* 7(3):287–296
2. Retrieved from http://elibrary.nusamandiri.ac.id/ebook/OpenSourceForYou-February_2018.pdf
3. Stajich JE (2002) The BioPerl toolkit: perl modules for the life sciences. *Genome Res* 12(10):1611–1618
4. Stajich JE (2007) An introduction to BioPerl. *Plant Bioinf*, 535–548
5. Gremme G, Steinbiss S, Kurtz S (2013) GenomeTools: a comprehensive software library for efficient processing of structured genome annotations. *IEEE/ACM Trans Comput Biol Bioinf* 10(3):645–656
6. Altschul S (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res* 25(17):3389–3402
7. Hubbard T (2002) The Ensembl genome database project. *Nucleic Acids Res* 30(1):38–41. <https://doi.org/10.1093/nar/30.1.38nio>
8. The GenomeTools Developer's Guide. (n.d.). Retrieved from <http://genometools.org/documents/devguide.pdf>
9. Taewan Ryu (2009). Benchmarking of BioPerl, Perl, BioJava, BioPython, and Python for primitive bioinformatics tasks and choosing suitable language. *Int J Contents* 5(2)
10. Talevich E, Invergo B, Cock PJ, Chapman BA (2012) Bio.Phylo: a unified toolkit for processing, analyzing and visualizing phylogenetic trees in Biopython. *BMC Bioinf* 13(1)

Protein Structure Prediction and Gene Expression Analysis

A Study on Protein Structure Prediction



Biboshan Banerjee, G. M. Siddesh and K. G. Srinivasa

1 Introduction

In an amino acid-chain molecule, the three-dimensional arrangement of atoms is known as a protein structure. Proteins are polymers; specifically, proteins are polymers of the polypeptides kind which are formed by sequences of amino acids, the monomers of polymer. A repeating unit of a polymer which indicates a residue is also known as a single amino acid monomer.

Proteins are polymers—specifically polypeptides—formed from sequences of amino acids, the monomers of the polymer.

When amino acids undergo condensation reactions in which one water molecule per reaction is lost in order for it to attach to one another with a peptide bond, a protein is formed. A chain which consists of less than 30 amino acids is often known as a peptide and not a protein [1]. Hydrogen bonding, van der Waals forces, ionic interactions, and hydrophobic packing are some of the non-covalent interactions which drive a protein into one or more specific spatial conformations. This happens so that proteins are able to perform their biological functions.

This is often necessary to determine the three-dimensional structure of a protein which helps us to understand the functions of the protein at a molecular level. The scientific field which employs techniques such as X-ray crystallography, NMR spectroscopy, and dual polarization interferometry to determine the structure of proteins is structural biology. The structure of proteins can vary in size from tens to several thousand amino acids [2].

B. Banerjee · G. M. Siddesh (✉)

Department of Information Science & Engineering, Ramaiah Institute of Technology, Bangalore, India

K. G. Srinivasa

Department of Informatics, Computer Science & Engineering, National Institute of Technical Teachers Training and Research, Chandigarh 160019, Chandigarh, India

While performing its biological functions, a protein can undergo a reversible structural change. This alternate structure is also referred to as a different conformational isomer simply conformations and the transition between them is called conformational changes.

2 Protein Structure Prediction

In the last couple of decades, the progress in the experimental determination of protein three-dimensional structure has been tremendous. But massive parallel sequencing technology has led to the rapid rise of sequence information and the experimental determination of protein three-dimensional structure has not kept pace with it. Therefore, the structure of more proteins is known than the three-dimensional structure of the protein. And the gap between the number of protein structures and the three-dimensional structure rather than diminishing is ever growing. Yet the information present in the amino acid of most proteins is enough so that the three-dimensional structure of those proteins can be determined, this opens up the possibility of predicting the three-dimensional structure of the protein from its amino acid sequence.

To bridge the widening gap, computational prediction of protein structures may be a viable option to the long-standing challenge of more than 40 years in molecular biology. Already, many accurate and useful three-dimensional models of proteins have been predicted by making use of the similarity of the amino acid of those proteins which are similar to another protein whose three-dimensional structure is already known. This is often called as template and homology model.

However, it is hard to predict the protein structure of protein when not even a single structure of a protein from its protein family is known. So how do we predict the structure of a protein?

There are three ways of predicting the structure of a protein.

- Comparative modelling
- Threading
- Ab initio prediction.

2.1 Comparative Modelling

Comparative modelling is based on the fact that proteins which are evolutionarily related and have similar sequences, which are found by checking the percentage of identical residues at each position established by an optimal structural superposition, have similar protein structure.

Comparative modelling is also known as homology modelling.

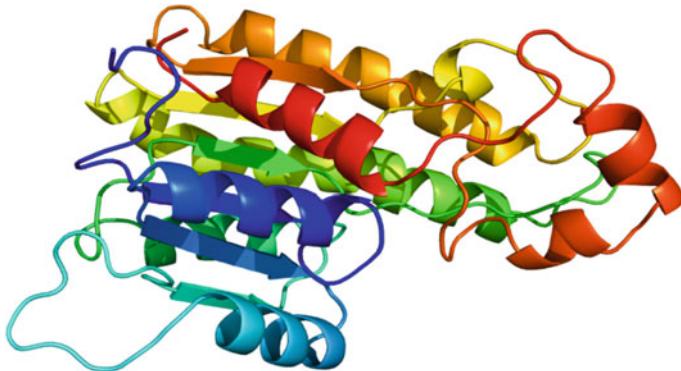


Fig. 1 Homology model of the DHRS7B protein

Homology modeling is the procedure to construct the atomic resolution model of the target protein, here target protein refers to the protein whose structure we want to predict from its atomic acid sequence, also an experimental 3D model is created which is known as a template from a related homologous protein (Fig. 1).

Homology modelling can be defined as the identification of 1 or more protein structures which will be similar to the structure of the target protein sequence. The second step in homology modelling is the production of an alignment that maps residues in the query sequence to residues in the template sequence. It has been observed that the protein structures which are homologues are more conserved to have similar protein structures, whereas protein sequences which have less than 20% sequence similarity may have vastly different protein structures [3].

Proteins which are related evolutionarily have known similar sequences are naturally homologous, and hence have similar structure. It has been observed that on the basis of sequence conversion alone, the predicted protein structure is evolutionary more conserved [4].

2.2 Threading

Threading is also known as fold recognition. Threading is the process of comparing the target protein sequence with a library of templates. The templates consist of the protein structure and fold data of various proteins. By comparing the target protein sequence with the templates, a list of scores is generated.

This type of modelling is done when the folds of a protein match the folds of protein whose structure is known, but the structures of the proteins are not homologous. Folding is the physical process in which a random coil folds into its characteristic and functional 3D structure from a polypeptide (Fig. 2).

Threading differs from homology in the sense that threading is done for the proteins which do not have homologous proteins present in the PDB.

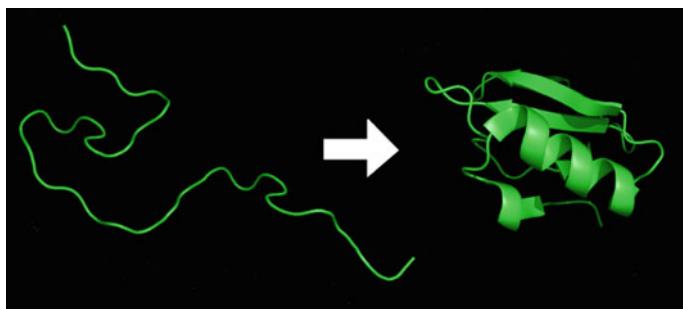


Fig. 2 Protein before and after folding

PDB: The Protein Data Bank (PDB) is a database which consists of three-dimensional structure data for a large number of a biological module. These biological modules can be proteins or nucleic acids. The structural data for the protein or nucleic acids are obtained by mainly cryo-electron microscopy; it is also obtained by X-ray crystallography and NMR spectroscopy. These structures are normally submitted by biochemists and biologists from all around the world into the PDB. The organization which manages the PDB is the Worldwide Protein Data Bank (wwPDB). In the last 3 years, all the new structures which have been submitted to the PDB have folds similar to the ones present in the PDB. Around 90% of the structures come under this category.

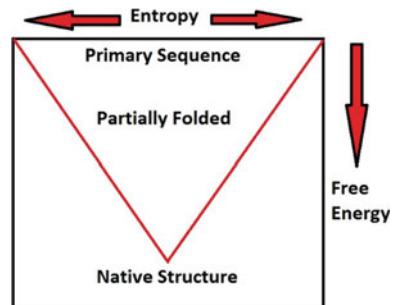
2.3 *Ab Initio Prediction*

In ab initio prediction is the method to find out all the energetics of the various processes which are required in the folding of the protein. And then finding the structures which has the lowest free energy out of all the structure we have modelled for that protein.

If there exists a protein whose tertiary structure is known to us and shares at least 30% of its structure with that of the protein whose structure is undetermined, then comparative methods can be used which will overlay the putative unknown structure with the structure of the known one and predict the structure of the unknown. But however, if there exists not protein which matches this threshold, then three other types of prediction must be applied to predict the structure of such a protein. They are ab initio protein prediction, fold recognition and threading.

- **Ab initio methods:** Physicochemical and neural net algorithms are used to derive the secondary structure of a protein from its primary structure. After this step, the algorithms predict the tertiary folding of the protein. This method is not capable of incorporating the locations and orientation of amino acid side chain.

Fig. 3 Free energy of native structure and partially folded structure



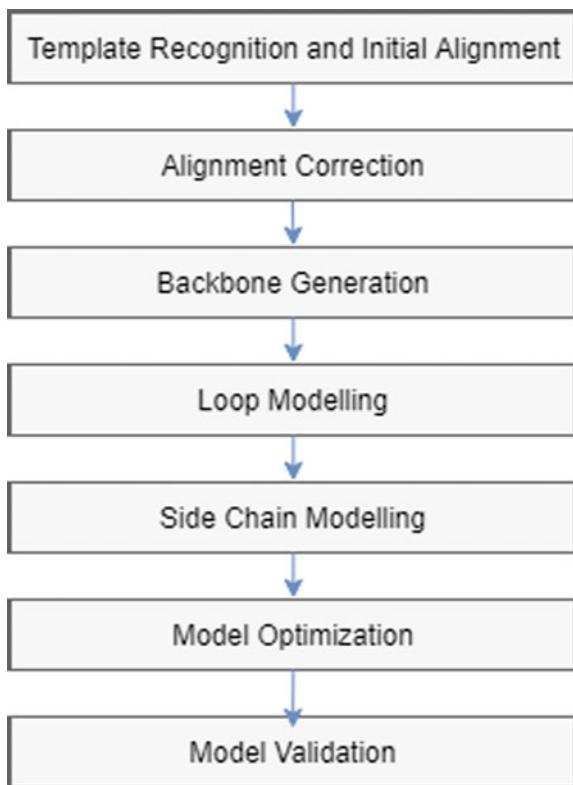
- Fold prediction: In fold prediction or recognition strategies, the secondary structure of the protein is first predicted, and then it is compared with libraries consisting of known protein secondary structure such as CATH or SCOP. And then the protein structure which seems like a match is assigned a confidence score.
- Threading: The fold recognition techniques are expanded further in threading. In threading, the unknown protein is placed onto a putative backbone of the protein which is the best fit, accommodating gaps where appropriate. The unknown protein is placed by empirically based energy functions for the interactions of residue pairs. Out of the interactions, the best interactions are chosen and then accentuated to discriminate amongst the potential decoys and predict the most likely confirmation.

From Fig. 3, we can clearly see that folded structure or native structures have lower free energy when compared to the free energy of the partially folded protein. Computer checks for this conformation as they indicate where the folding done is correct or not.

3 Homology Modelling

Homology modelling was done as early as the 1960s. Originally, models which were made of wire and plastic were used to display the bonds and atoms. The coordinates of proteins whose structure were known were taken and the models were constructed and the for those whose amino acids did not match the structure, the model was modified by hand. The first paper was published on 1969 by David Phillips, Brown and co-workers on homology modelling. This paper consisted of a modelled *a-lactalbumin* based on the structure of henegg white lysozyme. The homology between the proteins mentioned in the paper was 39%.

Example: Suppose we want to predict the structure of a protein A which is 150 amino acid long. So the first step will be to compare the sequence of protein A with all the sequences of the protein whose structures are known. These known structures are stored in the PDB. If there exists a protein B which is 300 amino acid long and consists of a region on amino acid which is 150 amino acid long which matches the 150 amino acid of A with at least 50% similarity which falls in the safe zone. Then,

Fig. 4 Homology steps

we can take a fragment from the structure of sequence B corresponding to the aligned region of that of sequence A. The amino acids that differ between the sequences A and B can be mutated to finally arrive at our model for structure A is called the target and is of course not known at the time of modelling.

Homology Modelling Steps (see Fig. 4).

3.1 Template Recognition and Initial Alignment

In this step, we search the PDB for homologous proteins which consists of determined structures. We can conduct this search using programs such as BLAST and FASTA because the percentage of the homology identity between the target sequence and a possible template should be high enough so it falls in the safe zone, which is one of the requirements of using these search programs.

BLAST Search

It is a search algorithm which can compare our query sequence with the sequences present in a library or database. And it gives us the sequences which match our query sequence. The sequence which are returned has a resemble above a certain threshold. The search can be varied depending on the sequence we want to query.

It can search amino acid sequence of protein, nucleotides of DNA and RNA sequences. It is developed by Stephen Altschul, Warren Gish, Webb Miller, Eugene Myers and David J. Lipman at the National Institutes of Health. So far, it has been cited 50,000 times. And it was published in the Journal of Molecular Biology in 1990.

FASTA Search

It is a software which is primarily used for DNA and protein sequence alignment. It takes our query sequence and searches a database for similar sequences using the local sequence alignment. It has a high speed of execution. First, it marks the potential matches using the pattern of word hits and word-to-word matches of a given length before performing a search using a Smith–Waterman type of algorithm which is a more time-consuming search.

It is also famous for its FASTA format which has become widely used in the bioinformatics domain. It is developed by William R. Pearson and David J. Lipman in 1985.

To attain the list proteins which are homologous with the target protein, the program compares the target query sequence with all the sequences of the known structures in the PDB using the below two matrices:

- A residue exchange matrix
- An alignment matrix.

Alignment correction

The alignment of two sequences of the two proteins where the percentage sequence identity is low can be difficult. One can then use the other sequences present from the homologous protein to find a solution.

The example is shown in Fig. 5a and b. We can see that it is nearly impossible for the sequence LTLTLTLT to be aligned with sequence YAYAYAYAY. So we find a third sequence TYTYTYTYT which can be easily aligned with both of them.

In Fig. 5, Model 2 is correct, because it leads to a smaller gap, compared to a huge hole associated with alignment 1.

4 Loop Modelling

After the sequence alignment step, due to insertion and deletion there are often regions created that lead to gaps in the alignment. Loop modelling is used to fill in

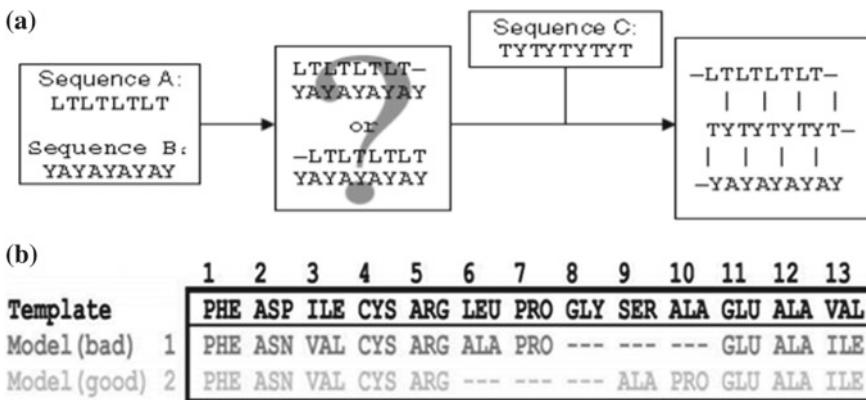


Fig. 5 **a** Sequence alignment example 1, **b** sequence alignment example 2

these gaps, but it is comparatively less accurate. The two main techniques are used to approach the problem:

- The database searching method: This method involves going through the database of known protein structures and finding loops and then superimposing these loops onto the two stem region of the target regions of the target protein. There exists some specialized software which can be used for this task, like FREAD and CODA.
- The ab initio method: In this method, various loops are generated randomly. And then, it searches of a loop in those loops which has reasonably low energy and φ and ψ angles which fall in the allowable regions of the Ramachandran plot.
- Side-chain modelling: It is important because is used to evaluate the protein–ligand interactions at the active sites and also the protein–protein interactions at the contact interface.

By searching every possible combination for every torsion angle of the side chain, we can select the one which has the lowest interaction energy with neighbouring atoms. A rotamer library can also be used, which has all the favourable side-chain torsion angles extracted from known protein crystal structures.

Model Optimization

Model optimization is done so that the overall conformation of the molecule has the lowest possible energy potential. This is done by adjusting the relative position of the atoms. The goal of this step is to relieve the steric collisions without altering the overall structure.

One other way of optimizing the model is by using molecular dynamics simulation. In molecular dynamics simulation, the atoms are moved towards a global minimum by applying various stimulation conditions like heating, cooling, considering water molecules. This leads to a better chance of finding the true structure.

$$\text{Energy} = \text{Stretching Energy} + \text{Bending Energy} + \text{Torsion Energy} \\ + \text{Non Bonded Interaction Energy}$$

Model Validation

Every homology model contains errors. Two main reasons are:

- The first error occurs if the sequence identity is more than 90% or less than 30% of the template and target protein. If the sequence identity is more than 90%, then it can be compared with the structures which are determined through crystallography. Else if it is less than 30%, then major errors can occur.
- The second major reason for errors is the errors present in the templates itself.

The factors for which the final model has to be evaluated are chirality, close contacts, stereochemical properties, bond length and $\varphi-\psi$ angles.

Advantages

- It can find the location of alpha carbons of key residues inside the folded protein.
- It can hypothesize structure functions' relationship which can be guided using this.
- Mutagenesis experiments can be guided using this.
- The putative active sites, binding pockets and ligands can be identified using the positions of the conserved regions of the protein surface.

Disadvantages

- The side-chain positions and predicting conformations of insertions or deletions are not possible with homology models.
- Homology models are only useful for drug designing and development process only if the sequence identity of the target protein is more than 70% with the template. Because template with less than 70% identity cannot be used in modelling and ligand docking studies which is necessary for this scenario.

Error sources in homology modelling

- Incorrect sequence alignment—among the most devastating error in homology modelling.
- Incorrect choice of template may happen, especially for multidomain proteins.
- The loop regions can be built incorrectly. Normally, the servers build the loops automatically. So, if the loop conformation being correct is important for us, then we need get the model built by multiple servers and then compare the models built by each of them.
- The person doing the modelling can also make some errors, but this type of errors can be hard to predict and it can include any kind of errors. One way to minimize this kind of errors is to have basic knowledge about the principles of protein structures.
- There can also be errors present in the template itself. This makes this type of errors difficult to deal with. A model can hardly be better than the template.

5 Use Case

Homology Modelling of Superoxide Dismutase

To start modelling using the SWISS-MODEL server, we need to first log into <https://swissmodel.expasy.org/>. Then click on Start Modelling (Fig. 6).

The server accepts four types of inputs: Sequences, Target Template Alignment, User Template and DeepView Project.

The protein sequence or the access code for the protein from UniProt is the most common way to give the input. The other input format it supports is uploading a file. If we want to give our own template for modelling, we need to select Target Template Alignment. If we want to upload a structure as a template, then we can choose User Template. A DeepView program user can choose the DeepView option.

We are going to use the Sequence Option and give the UniProt code of superoxide dismutase. The UniPort code of superoxide dismutase is P61851 (Fig. 7).

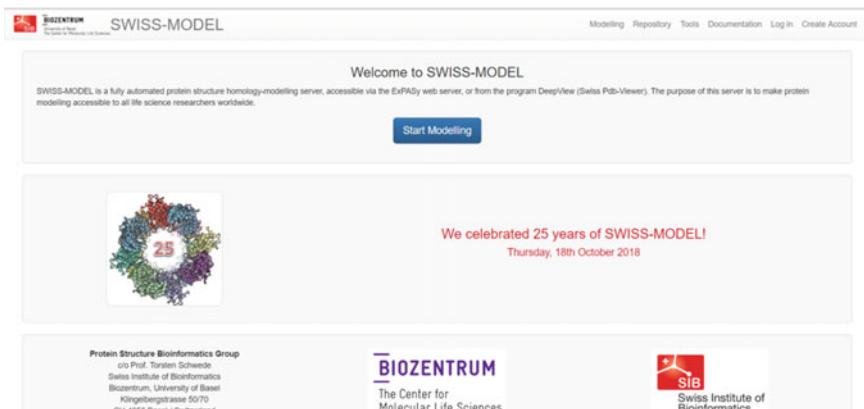


Fig. 6 Home page of SWISS-MODEL server [5]

Fig. 7 Starting a modelling project ad giving protein P61851 as the input [5]

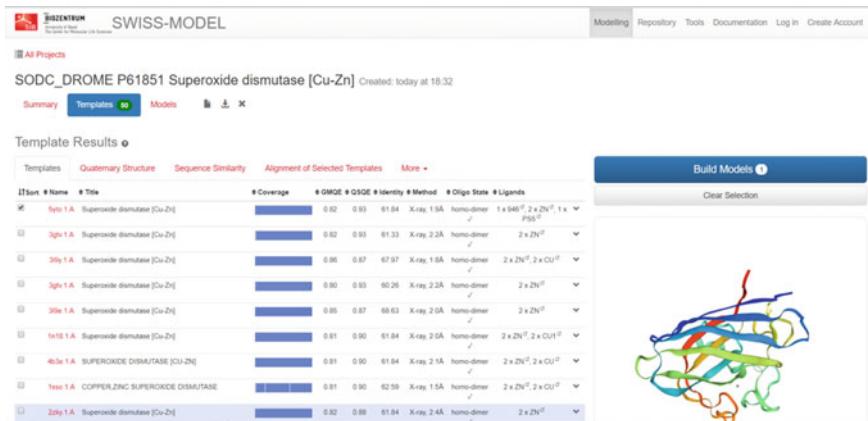


Fig. 8 All the templates available for our protein [5]

When they give the UniProt code, it automatically pulls the protein sequence from UniProt. Then, the next step is homology modelling to find good templates for our sequence. We can do that by clicking the “Search For Templates” button. The SWISS-MODEL server uses blast and HH Blitz to find suitable template. The template search takes less than 15 min.

Once the template search is done, we land on this page. It shows the template which is found for our protein. The results are sorted according to template score. We can also use the coverage row to select our templates. We can use the coverage row to learn a lot about the protein. Identity shows how many amino acids are exactly aligned, higher the identity the better. Method gives us the method with which the template was found and the resolution of the template. We must make sure to choose a template with high resolution (Fig. 8).

Subsequently, select the templates which be used for modelling which clicking on the check box, then click on the “Build Model” “button to build the models (Fig. 9).

After the model is finished build, we can see the model (Fig. 10).

Here, we can see all the details of our model. The GNQE values are updated compared to the values shown in the template selection page because it also takes the QMEAN value into account. The SWISS-MODEL uses the QMEAN score for quality estimation.

Homology Modelling Software (see Table 1).

Threading

Threading which also knows as fold recognition refers to a sequence-structure alignment approach used for proteins which do not have structural homology. It matches a protein sequence to a fold library using a so-called threading algorithm that assigns each amino acid to a position on the three-dimensional structure for a particular fold class which is available in the PDB. Energy minimization or modular dynamics is then used for structural refinement and selection of the best model.



Fig. 9 Details of the template chosen [5]



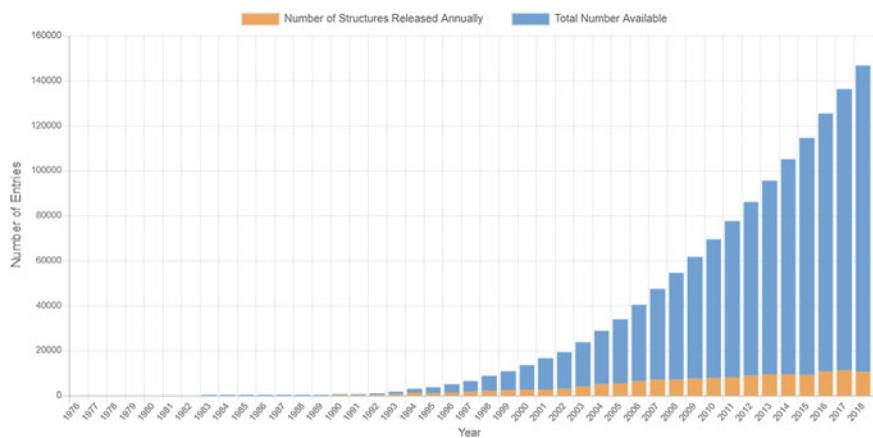
Fig. 10 Final built model [5]

It is similar to homology modelling in the sense that both the approaches try to build a structural model by using the experimentally solved structures as a template. But homology is only possible when the 3D structure of the protein is known. When the 3D structure is not known, then the path we take is threading (Fig. 11).

From Fig. 12, we can see growth of the protein data bank entries, right from the time the x-ray crystallographic analysis was done of myoglobin there has been a continuous submission of protein structure coordinates and over the last couple of decades there has been a steep increase in the structure and has come to reach a staggering number of more than one lakh forty thousand, but at the same time if you see the unique folds a protein can have we see in Fig. 8, it is not proportional to the number of structural coordinates that are being submitted.

Table 1 List of various homology modelling softwares

Software name	Method	Description
IntFOLD	It provides the interface for various programs such as prediction of protein–ligand binding residues, 3D model quality assessment, intrinsic disorder prediction, domain prediction, tertiary structure prediction/3D modelling	It is available as an online automated Web server and some of the program can also be downloaded
RaptorX	This program consists of different programs such as, remote homology detection, protein 3D modelling, binding site prediction	It is available as an online automated Web server and the program can also be downloaded
Biskit	It takes other programs and uses them in a automated manner	It consists of other programs like: BLAST search, T-Coffee alignment, and MODELLER construction
MODELLER	Satisfaction of spatial restraints	It's a standalone program mainly written in Fortran and Python
SWISS-MODEL	Local similarity/fragment assembly	It is available as an automated Web server (based on ProModII)

**Fig. 11** Overall growth of released structures per year [6]

For example, from Fig. 12, we can see that in the year 2010, we had 70,000 structures in the PDB, whereas from Fig. 8 we can see that in 2010, 1393 folds. In the year 2015 for 1 lakh structures, ten thousand we had the same number of folds, 1393.

Threading Steps (see Fig. 13).

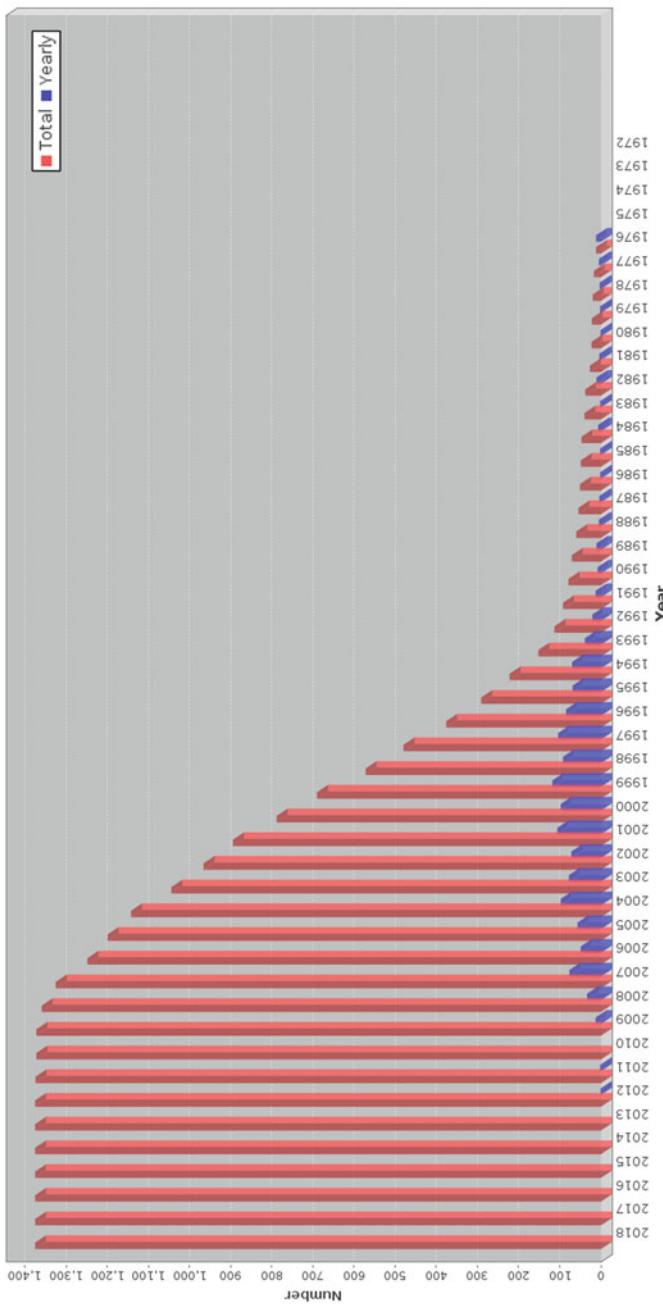
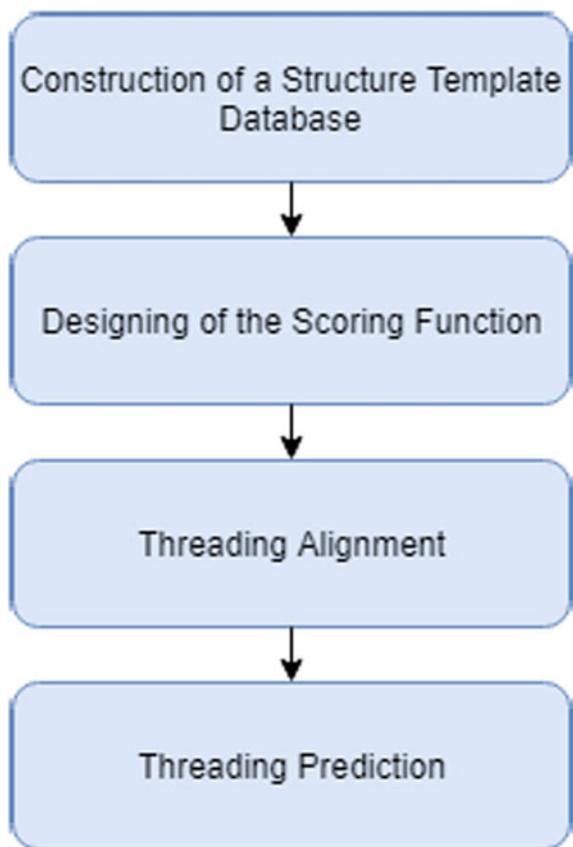


Fig. 12 Growth of unique folds per year as defined by CATH [7]

Fig. 13 Threading modelling steps



The construction of a structure template database:

To create a structure template database, we need to select the protein structures from a protein structure database. The selected protein structures are saved as structural template. When selecting the protein structure, we need to make sure that we avoid protein structures which have high sequence similarity. The protein structures can be selected from the various protein structure databases available on the Web such as PDB, FSSP, SCOP and CATH.

The design of the scoring function:

Mutation potential, environment fitness potential, pairwise potential, secondary structure compatibilities and gap penalties are the main things a scoring function should consist of. A scoring function which can be considered good should measure the fitness between the templates and the target sequence using the knowledge of the known relation of the structure and sequences. The prediction accuracy of the model is closely related to the quality of the energy function, especially the alignment accuracy.

Threading alignment:

After the scoring function is designed in the previous step, it has to be optimized so that the target sequence can be aligned with each of the structure templates. In every threading-based structure prediction program which take into account the pairwise contact potential, this is a significant task. Instead of using such a program, a dynamic programming algorithm can also be used for this task.

Threading prediction:

We should choose a threading alignment that is statistically most likely as the threading protein. A structural model is constructed by using the selected structural template and placing the backbone atoms of the target sequence on the structural template's aligned backbone positions.

Advantages and Disadvantages

Advantages

- It is moderately successful.
- It is good for proteins with less than 100 residues.

Disadvantages

- The methodology assumes that the current structural library consists of all the possible confirmation that could possibly be deciphered experimentally.
- Less than 30% of the predicted first are true remote homologs. The problem of identifying the best alignment is very challenging.
- High computational cost is involved to screen a library of thousands of possible folds.
- Energy functions are simplified for efficient calculations, and results are therefore compromised.

Threading Software (see Table 2).

Ab Initio Modelling

Ab initio means “from the beginning”, and the method predicts the structure of the protein from the scratch using sequence information by applying Newtonian force to achieve a thermodynamic stable native-like state. It is very useful in situations where the query sequence does not have any suitable templates in the protein structure library for either homology modelling; we also apply what we call as Anfinsen’s theory.

Anfinsen’s theory states that protein native structure corresponds to the state with the lowest free energy of the protein solvent system and that this particular information is contained in the amino acid sequence. The basic idea is to build empirical function that stimulates real physical forces and potentials of possible chemical contacts within the protein.

The methodology of ab initio is very dependent on Anfinsen’s theory. Anfinsen’s theory states that all the information of protein folding is contained in the sequence

Table 2 List of threading software

Name	Method	Description
HHpred	Template detection, alignment, 3D modelling	A popular threading Web server which run the HHsearch algorithm. It is mainly used for remote homology detection based on pairwise comparison of hidden Markov models
RaptorX	This program is much better than the previous version Raptor. The program in this software which are better is: remote template detection, single	Web server with job manager, automatically updated fold library
Phyre and Phyre2	Remote template detection, alignment, 3D modelling, multi-templates, ab initio	Web server with job manager, automatically updated fold library, genome searching and other facilities
FALCON	It does single-template and multi-template threading, a high-throughput server based on volunteer computing	Web server with job manager
MUSTER	Profile–profile alignment	It uses dynamic programming to run a standard threading algorithm and sequence profile to profile alignment. It also uses multiple structural resources to assist the sequence profile alignment

itself, and therefore, cell machinery is a material in protein folding. This holds true for most of the small globular protein. Protein native structure corresponds to a state with the lowest free energy for the protein solvent system, the protein energy landscape is very uneven, and it consists of many local minima in which partially folded proteins with relatively higher energy are seen. The global minima representing the native structure is found deep down the energy funnel passing through several molten globule states which is a number of intermediate conformational states between the unfolded states and the complete native state of the globular that are located there by you see the protein sequence at the very top intermediates in between which are called the molten globule state well folded protein or the global minima which is very close to the native structure. Having now understood that the proteins are just not amino acids that are strung together peptide bonds but hold vital information for the final protein structure. Software programs are so designed to use certain basic principles and concepts of physics to predict the three-dimensional structure of proteins (Fig. 14).

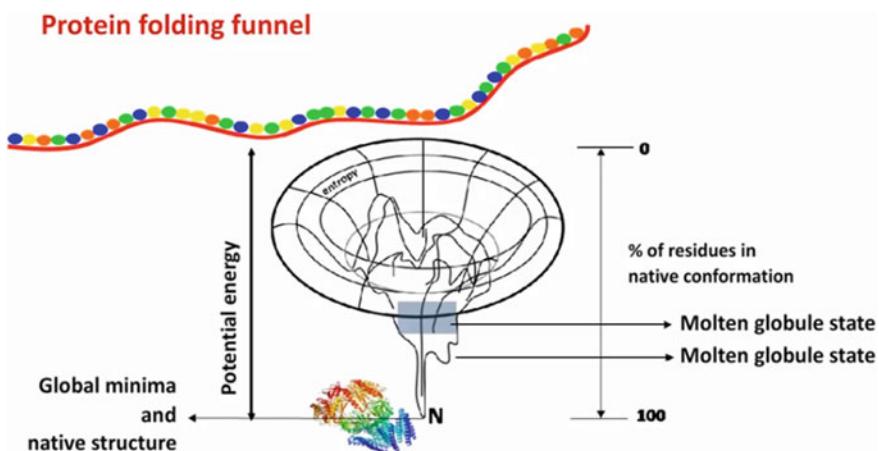


Fig. 14 Protein folding funnel

Ab initio Modelling Steps

- Protein representation: Apply rules and restrictions to be applied. Approaches to structure prediction differ with the assumption they make, and it can therefore be represented as three-dimensional coordinates in space or a set of dihedral angle pairs. Software differs in the way proteins are represented. The torsion angles are restricted to a finite set of values; hydrogen atoms are given importance in polar residues. Restrictions are laid for non-polar hydrogen atoms; only a finite set of dihedral angles is placed by single pseudo-atoms.
- Energy functions: It defines total potential energy of atoms. A potential energy functions specify the total potential energy of a system of all atoms as a function of their coordinates. These energy functions are basically a description of a set of functional forms and parameters used to compute the potential energy of the system of interest. These potentials are used as energy functions in the assessment of an ensemble of structural models generated during the modelling. The main parameters include bond stretching, angle bending dihedral torsion, H-bonding, van der Waals interactions and electrostatic interactions.

Physics-Based energy functions

Interactions between atoms are based on quantum mechanics and coulomb potential with only a few fundamental parameters such as the electron charge and the Planck constant that pertain to quantum mechanics. Compromised force field having a large number of selected atom types.

Well-known examples of such all atom physics-based force include AMBER, CHARMM, OPLS and GROMOS96.

We also have something called as solvation energy which reflects the implicit solvation model, the generalized bond (GB) model surface area-based model and the explicit solvation model TIP3P.

Energy potentials contain terms associated with bond lengths angles torsion angles, van der Waals and electrostatic interactions.

Knowledge-based static functions refer to the static energy potentials containing the empirical energy factors that are recognized from already reported protein structures in the protein data bank. They involve less computational cost; one of the most famous or well-known softwares that is used in this technique is I-TASSER.

This methodology could be either sequence independent which deals with atomic interaction potential, hydrogen bond potential, torsion angle potential, solvation potential, or it could be sequence dependent where pairwise residue contact potential, distance-dependent atomic contact potential, secondary structure propensities are made use of.

- Conformational search: Identify global minimum energy state using energy minimization; Monte Carlo; simulated annealing; MD or genetic algorithm.

The protein energy landscape is generally rocky; it contains many local energy minima and maxima along with intermediate saddle points, but it finally has only one global minima. The saddle points correspond to the transition states the barrier that all the molecules must cross if they are to fold to the native state. This is the point where the slope of the orthogonal function space becomes zero.

In order to find a valid ab initio structure model of a protein, it is important to have powerful conformational search method which for a given energy function can accurately and efficiently find the global minimum energy structure. The different methods one could possibly use are as follows:

1. Energy minimization

It is a molecular/quantum mechanical process to find the lowest energy conformation of a protein.

Energy minimization methods:

- a. First-order minimizations: steepest descent, conjugate gradient
- b. Second derivative methods: Newton-Raphson method
- c. Quasi-Newton methods.

2. Monte Carlo is an important optimization and sampling technique that does not force but rather compare energies. It makes use of Boltzmann probabilities.
3. Molecular dynamics simulation aimed to understand the properties of assemblies of molecules and the atomic-level interactions between them. Software: CHARMM and NAMD.

Model selection: Choose the best native-like structure from pool of decoy structures.

Irrespective of conformational method used, we arrive at a number of possible decoy structures and then need to choose the final one; the methods that could possibly be used are based on energy or clustering.

4. Monte Carlo is an important optimization and sampling technique that does not force but rather compare energies. It makes use of Boltzmann probabilities.
5. Molecular dynamics simulation aimed to understand the properties of assemblies of molecules and the atomic-level interactions between them. Software: CHARMM and NAMD.

Model selection: Choose the best native-like structure from pool of decoy structures.

Irrespective of conformational method used, we arrive at a number of possible decoy structures and then need to choose the final one; the methods that could possibly be used are based on energy or clustering.

In the energy-based methods, consider specific potentials and identify lowest energy state, for example as the one used in software ASTRO-FOLD. In the other method of clustering, the cluster centre conformations of the largest cluster are considered as the one closest to the native structure. The software that best uses the clustering method is Rosetta.

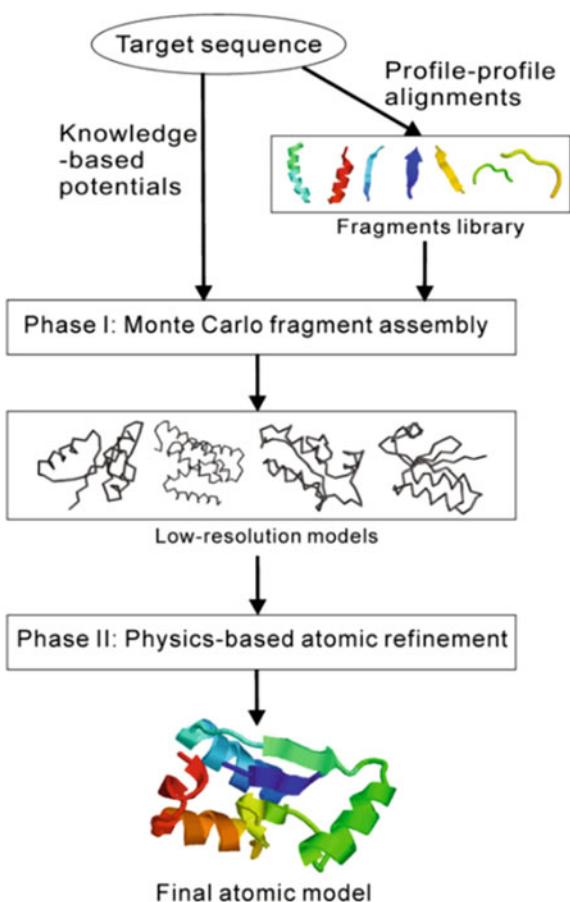
Below, we can see the steps followed in Rosetta and I-TASSER modelling. These are the most common and widely used ab initio modelling algorithms.

In Fig. 15, we can see the various steps of the Rosetta protocol. In the first step, the target protein is split into fragments. The target proteins are unrelated to the protein structure in the PDB. The proteins in the PDB are the ones which are used to create the full-length models by simulations which guided by a knowledge-based force field. The second step is used only for refining the models selected in the first step. The models are refined at an atomic level using physics-based potential.

In Fig. 16, we see the data flow of the I-TASSER algorithm. First, various threading programs are used to identify the templates and the super-secondary structure fragments. Full-length models are reassembled using the segments which are excised from the continuously aligned regions. Also, in the full-length model the threading aligned regions are built using the simulations based on lattice-based ab initio simulation. The next step in the algorithm is to search for templates which are similar to the templates found in the first step from the PDB by structure alignment. To assist in the second-round refinement, the spatial restraints are extracted from the templates. In recent times, several developments were made to improve the results of distant homology modelling such as sequence-based contact predictions and segmental threading (Table 3).

Ab initio Modelling Software (see Table 4).

Fig. 15 Flow chart of the Rosetta protocol [8, 9]



6 Conclusion

In this chapter, we have discussed various computation methods for protein structure prediction like comparative modelling, threading and ab initio methods. It has been observed that discussed methods have their own advantages and disadvantages. Also, the selection of methods is purely based on the data properties like mutual information or simple correlation coefficients which are sufficiently indicative of the relevance of the features.

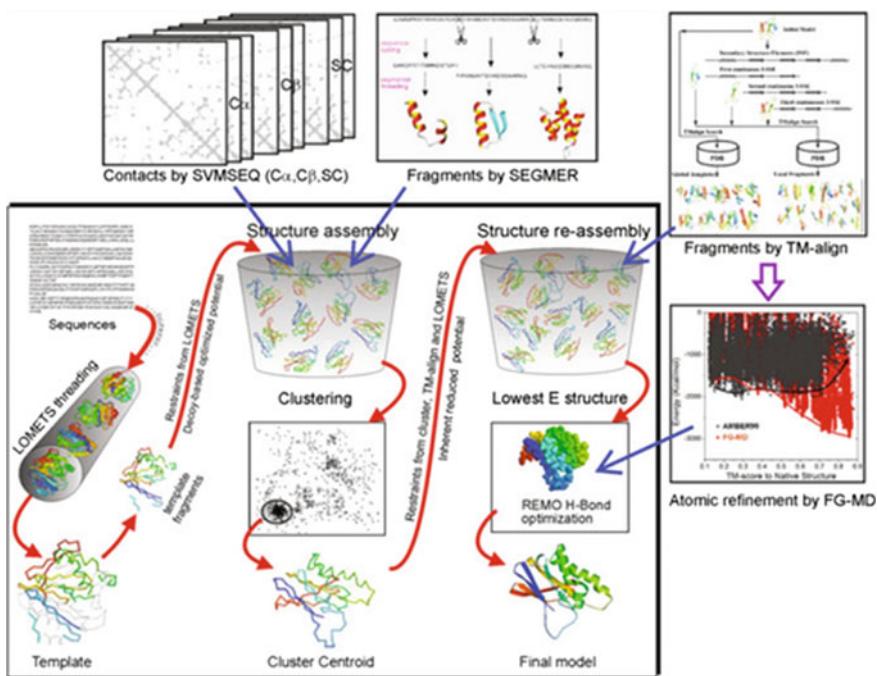


Fig. 16 Flow chart of I-TASSER protein structure modelling [8, 10]

Table 3 A list of ab initio modelling algorithms along with their energy functions, conformational search methods, model selection schemes and typical CPU time per target

Algorithms	Force field	Search method	Model selection	Time cost per CPU
AMBER/CHARMM/OPLS	Physics-based	Molecular dynamics (MD)	Lowest energy	Years
UNRES	Physics-based	Conformational space annealing (CSA)	Clustering/free energy	Hours
ASTRO-FOLD	Physics-based	CSA/MD	Lowest energy	Months
ROSETTA	Physics and knowledge based	Monte Carlo	Clustering/free energy	Days
TASSER/chunk-TASSER	Knowledge based	Monte Carlo	Clustering/free energy	Hours
I-TASSER	Knowledge based	Monte Carlo	Clustering/free energy	Hours
QUARK	Physics and knowledge based	Monte Carlo	Clustering/free energy	Hours

Table 4 List of ab initio modelling software

Name	Method	Description
EVfold	It uses the correlated mutations in a protein family to calculate the evolutionary coupling. Also, it can use only the sequences to predict the 3D structure and use the coupling strength from functional residues. It can also predict both globular and transmembrane proteins	Web server
FALCON	Uses a position-specific hidden markov model for predicting the protein structure. It does this by refining the distributions of the dihedral angles iteratively	Web server
Rosetta@home	The Rosetta algorithm implementation in a distributed computing format Distributed computing implementation of Rosetta algorithm	Downloadable program
Robetta	Rosetta homology modelling and ab initio fragment assembly with Ginzu domain prediction	Web server
Selvita protein modelling platform	Package of tools for protein modelling	It is a Web server and a standalone program which include the CABS ab initio modelling

References

1. Stephen Stoker H (2015 Jan 1) Organic and biological chemistry. Cengage Learning, p 371. ISBN 978-1-305-68645-8
2. Brochieri L, Karlin S (2005-06-10) Protein length in eukaryotic and prokaryotic proteomes. Nucleic Acids Res 33(10):3390–3400. <https://doi.org/10.1093/nar/gki615>. pmc 1150220. pmid 15951512
3. Chothia C, Lesk AM (1986) The relation between the divergence of sequence and structure in proteins. EMBO J 5(4):823–826. PMC 1166865. PMID 3709526
4. Kaczanowski S, Zielenkiewicz P (2010) Why similar protein sequences encode similar three-dimensional structures? Theoret Chem Acc 125(3–6):643–650. <https://doi.org/10.1007/s00214-009-0656-3>
5. <https://swissmodel.expasy.org/>
6. <https://www.rcsb.org/stats/growth/overall>
7. <http://www.rcsb.org/pdb/statistics/contentGrowthChart.do?content=fold-cath>

8. https://www.researchgate.net/figure/Flowchart-of-the-ROSETTA-protocol_fig1_225193759,
https://www.researchgate.net/figure/Flowchart-of-I-TASSER-protein-structure-modelling_fig3_225193759
9. Simons KT, Kooperberg C, Huang E, Baker D (1997) Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions. *J Mol Biol* 268(1):209–225
10. Lee J, Wu S, Zhang Y (2009) Ab initio protein structure prediction. In: Rigden DJ (ed) From protein structure to function with bioinformatics. Springer, Dordrecht

Computational Methods Used in Prediction of Protein Structure



Poulami Majumder

1 Introduction

Protein is the basic building block of life. This is the key component of the body which is responsible for various physiological biochemical reactions. Protein is an important chunk in bioinformatics field to understand the possible biological process of life. It is very important to predict the protein structure to pursue the following challenges like drug design, medicinal application as well as in bio-industrial applications [1]. Over 25 years, the way out towards prediction of protein structure has been continued. Multiple kinds of approaches have been taken for protein structure prediction through computational approaches which have been developed as the most popular and useful in recent times [2]. To know about the different protein structure prediction approaches, we should first focus on the overview of the structure of protein.

Four types of protein structure have been discovered so far (Fig. 1). Those are primary structure, secondary structure, tertiary structure and quaternary structure [3]. The primary structure of protein is based on the simple linear arrangement of amino acid residue sequences [4]. The secondary protein structure is generally based on the binding pattern of the amino hydrogen and carboxyl oxygen atoms between amino acid sequences throughout the peptide backbone [5]. There are two kinds of protein secondary structure, and those are alpha helices and beta strands. In these structures, the amino acid sequences are linked with each other by hydrogen bonds. The alpha helix structure is generally composed of 3.6 amino acids per turn along with hydrogen bonds which are formed between every fourth residue while in beta strands there are two portions of the chain—one is upward with 5–10 consecutive amino acids and another is downward 5–10 consecutive amino acid sequences. H-bond interactions are formed mostly in between adjacent amino acids and short loops

P. Majumder (✉)

Department of Biotechnology, Maulana Abul Kalam Azad University of Technology, Kolkata, West Bengal 700064, India

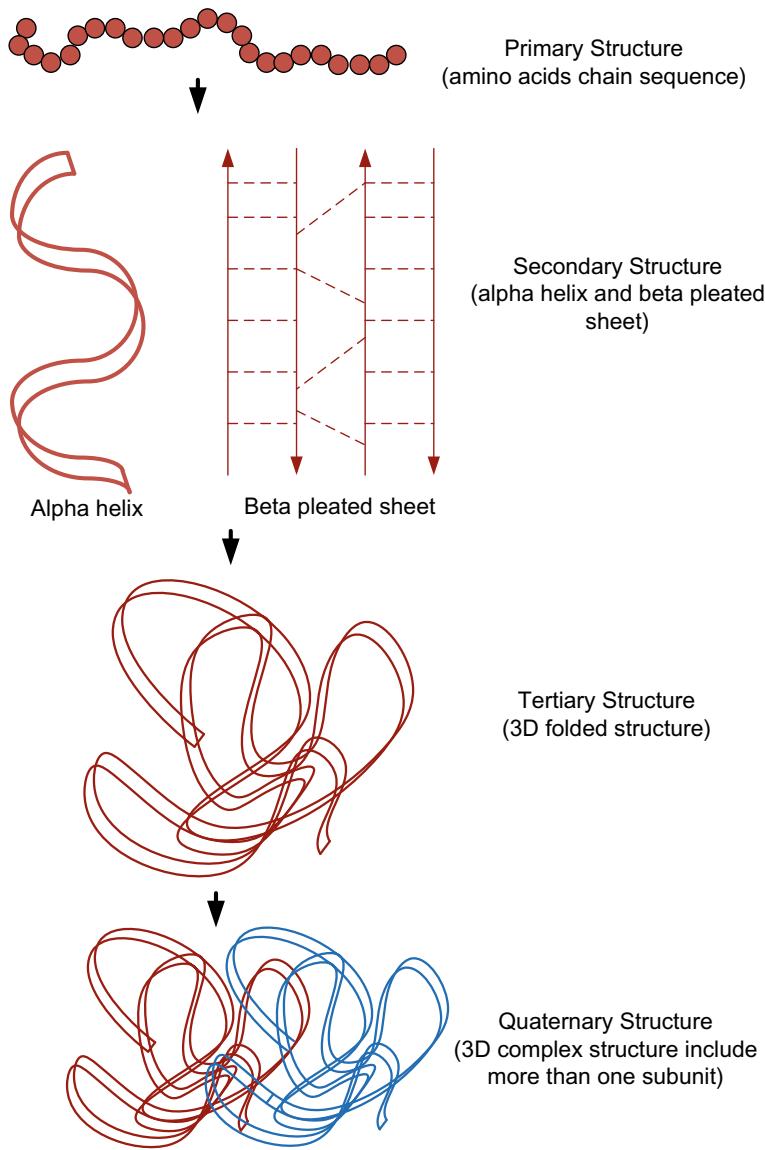


Fig. 1 Four levels of structure

between them [6, 7]. This secondary structure prediction is more likely related to the pattern of alpha helices and beta strands amino acids residue structure. The prediction of secondary structure is mainly focused on to know about the linear amino acid sequences, i.e. primary protein structure [8]. The pattern of the amino acid residues arrangements, their size and shape directs the ligands to fit with the protein in a better

way. The tertiary structure is about the three-dimensional structure of monomeric and multimeric molecules [9]. In this structure, alpha helix and beta strands formed a globular structure together. The folding structure of this kind of protein is initiated by the hydrophobic bond, di-sulphide bond, salt bridge and also H-bond. The texture of this structure is not so rigid, as it is fluctuated minutely in continuous manner. The quaternary structure is built up through dimeric and/or multimeric molecules stabilized by the non-covalent bonds. The structural annotation of a protein is key understanding towards the function of a protein [10]. It is also an important thing to know whether the structure of a protein is in its correct conformation or not, if so then is that correct conformation results the efficient function. The pattern of amino acid residue sequences determines the protein structure. Generally, a rough sequential structure of amino acid residues is the key to predict the complex protein structures. However, this experimental prediction is hard to find about the particular function of proteins [11].

The knowledge about the primary structure, i.e. linear amino acid sequence, is not enough as the conformational configuration is getting fluctuated continuously. Recently, a number of techniques have been developed to determine the three-dimensional structure of protein, namely electron microscopy, spectroscopy, X-ray crystallography and nuclear magnetic resonance (NMR) [12]. However, there is a wide technical slit between the known sequential structure and the predicted structure that has been found which is also a challenge towards protein structure prediction. Computational method is to resolve the protein structure prediction challenges directly from the amino acid sequences. In this book chapter, the major computational tools or approaches have been described for protein structure prediction along with their different software programs available in the market. This chapter aims to deliver an overview on computational methods used in protein structure prediction.

2 Computational Methods for Protein Structure Prediction

Three major strategies of computational method have been taken to predict the protein structure and those are as follows:

- Homology modelling techniques or comparative techniques,
- Protein threading or protein fold recognition and
- Ab initio or de novo techniques.

In Fig. 2 the basic concept of protein structure prediction has been illustrated schematically based on different protein modelling stated earlier.

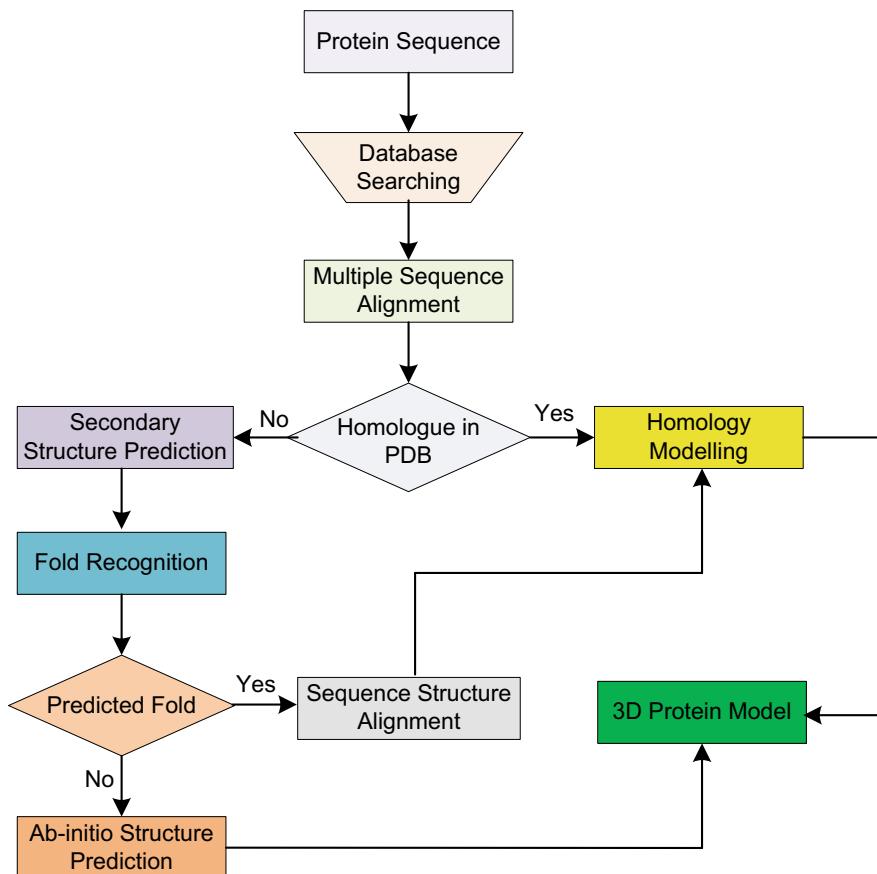
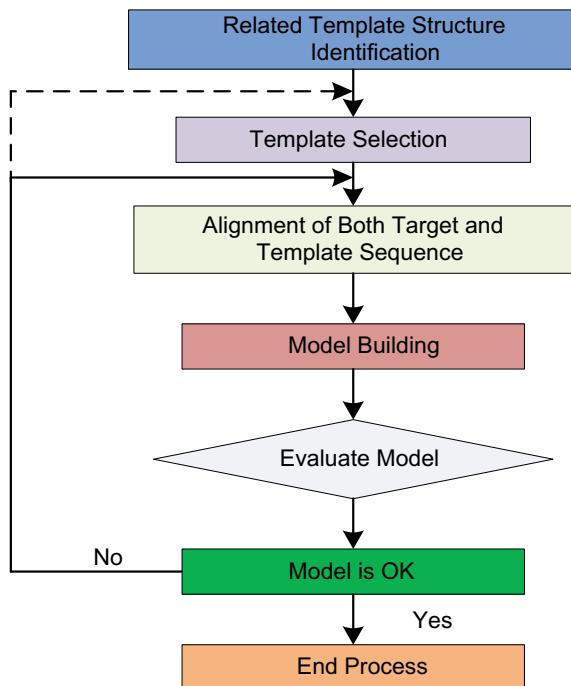


Fig. 2 Decision-making chart for protein structure prediction method

2.1 Homology Modelling Techniques

This technique helps to paradigm an unknown atomic-resolution model of the “target” protein retrieves from its amino acid sequence which is followed by the formation of experimental 3D structure of a related homologous protein [13]. This technique identifies one or more known protein structures which are similar to the required protein structure and aligns both the sequences of known and unknown proteins to match them at its best. In this technique, one can also predict an unknown protein structure based on multiple templates which are used for different parts of protein [14]. The structural accuracy generated by homology modelling is highly reliable based on the amino acid sequences resemblance between the target (queried protein) and template (existing known protein) protein. The generated models are thought to be reliable if the sequence resembles more than 50%. In some cases, the resemblance is less than 20% [15]. Hence those protein structure predictions need further techniques other

Fig. 3 Schematic illustration of basic process of homology modelling for protein structure prediction



than homology modelling. This modelling is useful in the pharmaceutical industry to structure-based drug discovery and drug design [16].

This process is comprised of following methods (Fig. 3):

- template selection;
- amino acid sequence alignment between template and target protein;
- alignment correction and model backbone construction;
- side chain generation and optimization;
- overall model optimization, assessment and verification.

A number of homology modelling techniques are available in the market, and most of them are open source. In Table 1, some significant programs with their significant description and function have been described. In this table, some significant homology models are highlighted. However, there are many other models which are used throughout the world such as IntFOLD, GeneSilico, Geno3D, STRUCTUROPEDIA and WHAT IF.

2.2 Protein Threading

Protein threading is nothing but protein fold recognition. In this technique, the known proteins with same fold are being used as template for modelling the target protein

Table 1 Some useful computational methods based on homology modelling techniques (Courtesy Wikipedia) [17]

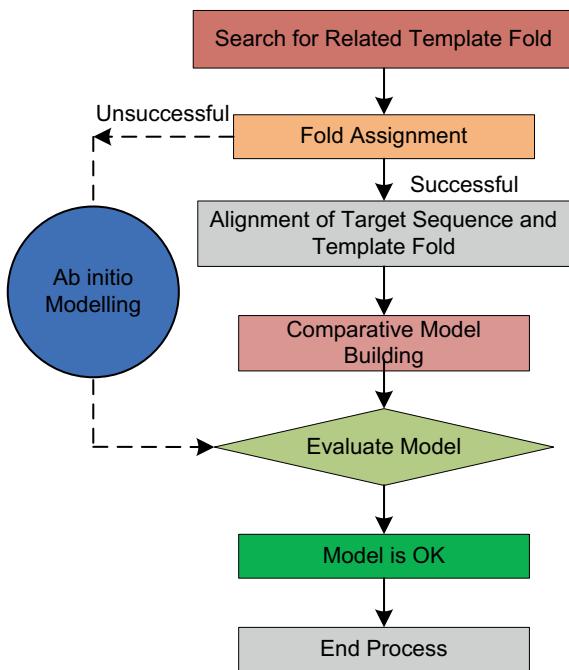
Name	Description/function
RaptorX	One of the most popular methods. It does protein 3D modelling, detection of remote homology and the prediction of binding site
Biskit	It is an open-source software package programmed in Python. It wraps external programs into automated workflow
ESyPred3D	It is an automated homology program that helps to predict template sequences, alignment and 3D modelling. It is majorly focused on alignment strategy
FoldX	It uses empirical force field to design algorithm for protein structure. Energy calculations and protein design are built
HHpred	It is an open-source software package. Template detection, alignment, 3D modelling of sensitive protein structure
MODELLER	This model is used to build tertiary and quaternary protein structure
Phyre and Phyre2	Free web-based service. It is one of the popular methods which helps in residues alignment, remote template detection and 3D modelling by using multiple templates
Prime	It works on sequence alignment, secondary structure prediction, homology modelling, protein refinement, loop-prediction and side chain prediction
Bhageerath-H	This platform was established by IIT Delhi and mainly focuses on tertiary protein structure prediction
SWISS-MODEL	This homology modelling is currently most accurate method for protein structure prediction. It works by finding the local similarity and fragment assembly
YASARA	Detection of templates, hybridization of model fragments, alignment, ligands and oligomers designing

[18]. There is a minute difference in protein threading from protein homology modelling. Protein threading specifically targets the protein with same fold level that means it aligns the sequence to the template structure while homology modelling is for comparatively easier target which aligns the sequence to the template sequence only [19]. There are specific interactions between the amino acid sequences that affect the protein folding like hydrogen bond, hydrophobic bond, Van der Waals interaction, electrostatic force, etc. [20]. There are almost 1300 different known protein folds which are existing till now, though each year new folds are being discovered. Protein threading is a process that comprises of four major steps (Fig. 4).

Those are as follows:

1. Library of core fold templates which represents the template structures (protein data bank database).
2. The compatibility between the aligned amino acid sequences and template fold including the compatibility evaluation.
3. Search for the best option to optimize the target sequence and the template structure alignment.
4. Evaluate the best match based on statistical significance.

Fig. 4 Schematic representation of protein threading in simpler way



In Table 2, some significant usable protein threading software is mentioned. These help in computational modelling of target protein based on template fold by fold recognition method. There are plenty of computational algorithms that have been proposed to find the best optimal protein threading of sequences onto a structure, but finding the best template for alignment is still difficult due to very limited resource in PDB, though researchers are trying many combinatorial optimized approaches such as simulated annealing, conditional random fields, branch and bound and linear programming. If homology modelling has been performed to predict protein structure and the aligned sequence is very low (<25%), then in that case protein threading could be a better use for good compatible prediction [21–23].

2.3 Ab Initio Modelling

Predicting the nature of protein structure from its amino acid sequences is really a tough job. We have discussed earlier about protein homology modelling and threading. These two popular methods are used for protein structure prediction based on the sequence and/or structural fold compatibility [24]. But if homologs do not exist in the resource or existing homologs cannot be identified, then another way out must be found. In this case, ab initio modelling has been found. It is used to predict comparatively complex protein structure such as tertiary structure [25]. This method requires

Table 2 Some useful computational methods based on protein threading (Courtesy Wikipedia [17])

Name	Method
HHpred	Popular protein threading software, it helps in template fold detection, alignment, 3D modelling based on pairwise comparison of hidden Markov models
RaptorX	Single and multiple template threading, remote template detection
Phyre and Phyre2	Remote template fold detection, sequence alignment, multi-template threading, 3D modelling
MUSTER	It is a protein threading algorithm. It is based on sequence profile-to-profile alignment and dynamic programming along with multiple alignments
BioShell	It is a protein threading algorithm. It is using adjusted profile-to-profile dynamic programming algorithm shared with predicted secondary protein structure
SPARKS-X	It works statistically, purely based on probability. It makes the sequence to structure match of target protein and template protein. 3D structure modelling done according to the sequence and structural profiles
Building Blocks Structure Predictor (BBSP)	Hybrid template-based
DeepFR	Remote template selection, 3D modelling

a large number of computational resources to predict the structure of complex one. This modelling is very useful in the research field of medicine and drug design [26]. In this modelling, a conformational search has been done based on the designed energy function which generates possible compatible conformations (decoy structure) and the most suitable one should be picked up. Ab initio modelling generally depends on three key factors, those are as follows: energy function design, conformational search engine and model selection strategy [27–30].

- Energy function design: A possible decoy structures have been generated but the most suitable one is being chosen on the basis of thermodynamically stabled native protein. An accurate energy function has designed with which that native protein structure contacts in a condition. This is classified into two groups: (a) physics-based energy function and (b) knowledge-based energy function. In case of physics-based energy function, the atom's alike physical and chemical properties are calculated while knowledge-based energy function statistically solves the protein structure [31, 32].
- Conformational search engine: It is an efficient search method. It can quickly identify the low-energy states through conformational search. There are two

popular computational methods which are good in search of conformational space those are Monte Carlo (MC) and molecular dynamics (MD). MC and MD both need large computational resources. Another two categories are genetic algorithm and mathematical optimization. MC simulates the random sample parameters to explore the complex structural behaviour. MD simulation is being used to understand the movement of atoms of protein. Genetic algorithm solves the natural selection problems by repeatedly modifying the population while mathematical optimization selects the best element set from the pool of available related alternatives [33–35].

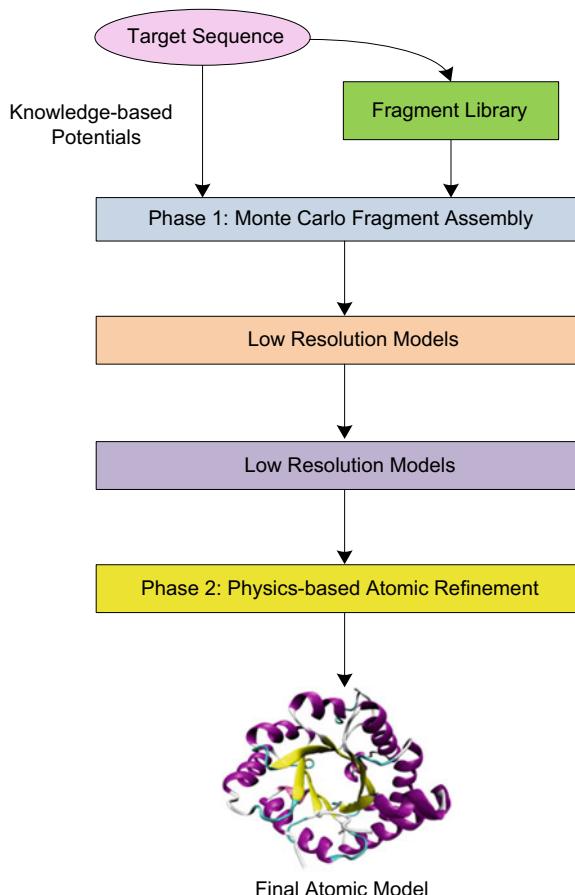
- iii. Model selection strategy: That can select near-native models from a pool of decoy structures. It is both energy-based and free energy-based. It helps to select the decoy with the lowest energy [36].

The initial strategy of ab initio modelling is to elucidate the secondary protein structures from its primary structure, i.e. linear amino acid sequences, and then it is followed by the tertiary structure prediction based on physicochemical parameters. Whenever the structure prediction cannot be done by homology modelling or threading, ab initio will resolve the problem generally. However, this modelling has limitation related to the exploration of locations and orientation of amino acid side chains [37]. Another major limitation of this modelling is very time consuming to get a successful solution [38]. In Table 3, some of the significant methods are enlisted which are widely used in ab initio modelling. In Figs. 5 and 6, the two kinds of ab initio modelling—Rosetta and I-TASSER—have been schematically illustrated by showing the flowchart of the whole method [39, 40].

Table 3 Some useful computational methods based on ab initio modelling (Courtesy Wikipedia [17])

Name	Method
FALCON	It predicts protein structure by position-specific hidden Markov model through refining the dihedral angles distribution
QUARK	It obeys the Monte Carlo method for complementary conformation search and fragment assembly
I-TASSER	One of the popular ab initio modelling. It involves protein threading followed by fragment structure reassembly
Rosetta@home	It designs new proteins by predicting the nature of protein–protein docking with the help of about sixty thousand active volunteered computers. Rosetta algorithm has been implemented in distributed-computing system
ROBETTA	Combination of ab initio fragment assembly and Rosetta homology modelling along with Ginzu domain prediction
Bhageerath	A computational protocol for modelling and predicting protein structures at the atomic level
Abalone	It aims to predict protein folding and DNA-ligand complexes. It simulates biomolecules by molecular dynamics folding and molecular graphics program

Fig. 5 Flowchart of Rosetta protocol



2.4 CASP

CASP is an advanced widely used process of protein structure prediction. It is ab initio modelling-based advanced protein structure prediction method. It is a distinguished significant process and that is why it has been discussed separately in this different section. CASP stands for Critical Assessment of protein Structure Prediction which provides an independent analysis on state of the art in protein structure modelling which helps the large researcher community to get an immense idea about protein structure [41]. It is a double-blinded method where initially the target sequences have been predicted through X-ray diffraction or NMR method and those sequences are to be registered in the modelling community, and then models are analysed by the automated method or independent assessors [42]. Since last 20 years of the CASP modelling, the structure modelling field has changed rapidly. In protein data bank (PDB), only 229 unique protein folds were known which is very less [43]. Most of

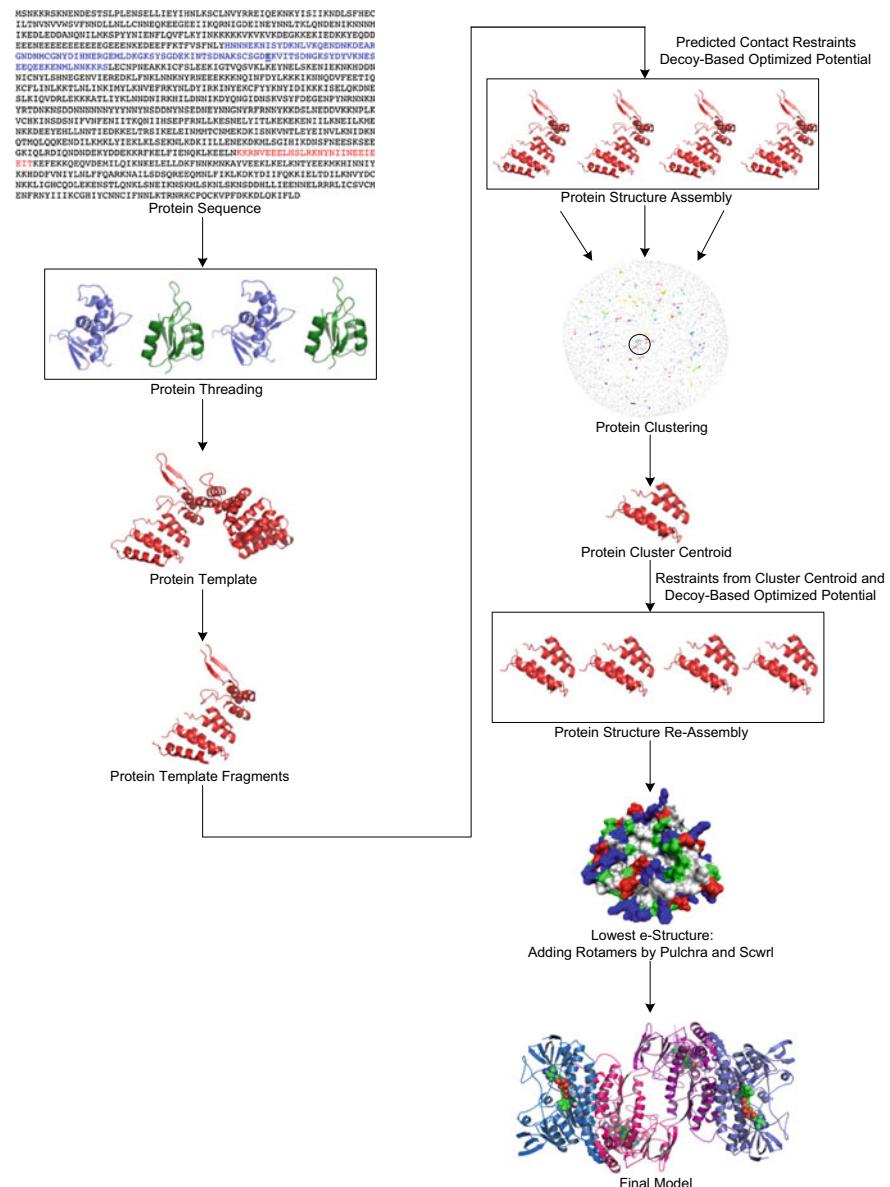


Fig. 6 Flowchart of I-TASSER protein structure modelling

the target sequences were not in the PDB, so it is difficult to detect the sequences through homology modelling or threading. Ab initio modelling could be the best alternative for those cases. CASP provides the overall understanding regarding the accurate model and amino acid sequences. Even the homology modelling of protein has been monitored and can be improved by CASP experiment.

There are lots of CASP experiments which are used in different stages of protein structure prediction at different levels. Those are as follows [44–46]:

All types of CASP can predict tertiary structure

- CASP5 can predict secondary protein structure and also detect the disordered regions.
- CASP6 helps in protein function prediction.
- CASP7 assesses model quality and model improvement
- CASP8 successfully matches the full template model with the target with accuracy.
- CASP10 evaluates the binding sites nature.
- CASP ROLL aims to predict larger number of targets.

There are some other protein structure prediction computational methods which are used in secondary protein structure prediction, transmembrane protein and signal peptide predictions. In Table 4, those computational methods are enlisted.

Table 4 Computation methods for different kinds of protein structure prediction (Courtesy Wikipedia) [17]

Name	Method description	Type
Porter 5	Fast, state-of-the-art ab initio prediction of protein secondary structure in 3 and 8 classes	Secondary protein
RaptorX-SS8	Predicts both 3-state and 8-state secondary structures by using conditional neural fields from PSI-BLAST profiles	Secondary protein
GOR	Information theory/Bayesian inference	Secondary protein
Jpred	Multiple neural network assignment from PSI-BLAST and HMMER profiles. Predicts secondary structure and solvent accessibility	Secondary protein
PredictProtein	Profile-based neural network	Secondary protein
PSIPRED	Two feed-forward neural networks that perform analysis of the PSI-BLAST based output	Secondary protein
HyperChem	Frequency analysis of amino acid residues observed in proteins	Secondary protein
HCAM	Hidropathy Clustering Assisted Method by detection of physicochemical patterns	Secondary protein
HMMTOP	Hidden Markov model	Transmembrane helix
PHDhtm	Multiple alignment-based neural network system	Transmembrane helix
SignalP	Artificial neural networks and hidden Markov models	Signal peptide

3 Conclusion

This chapter envisages the overview of the available significant computational methods and software to predict the structure of protein. Amino acid sequences are the backbone of all protein structure. However, it is not enough to know about the name of amino acids only. The pattern of amino acid sequences, the type of folding, conformational space, concentration, free energy pH, active binding sites, etc. are also very important to know to predict the function as well as the state of queried protein based on its structural basis. Protein structure prediction fulfils the gap between the amino acid sequences and protein structure. The prediction method is limited to the protein homology, threading and ab initio modelling; however, their successful prediction helps many researchers to bloom their interest in proteomics and the budding number of servers and groups taking part in community-wide prediction quality assessment experiments are the resilient of that. Though it is necessary to make more progress in development of remote sequence detection as well as the accurate identification of novel protein sequences in all aspect.

References

1. Venkatesan A, Gopal J, Candavelou M, Gollapalli S, Karthikeyan K (2013) Computational approach for protein structure prediction. *Healthc Inform Res* 19(2):137–147. <https://doi.org/10.4258/hir.2013.19.2.137>
2. Broccieri L, Karlin S (2005) Protein length in eukaryotic and prokaryotic proteomes. *Nucleic Acids Res* 33(10):3390–3400. <https://doi.org/10.1093/nar/gki615>
3. Sanger F, Tuppy H (1951) The amino-acid sequence in the phenylalanyl chain of insulin. I. The identification of lower peptides from partial hydrolysates. *Biochem J* 49(4):463–481. <https://doi.org/10.1042/bj0490463>
4. Perticaroli S, Nickels JD, Ehlers G, O'Neill H, Zhang Q, Sokolov AP (2013) Secondary structure and rigidity in model proteins. *Soft Matter* 9(40):9548–9556. <https://doi.org/10.1039/C3SM50807B>
5. Nickels JD, Perticaroli S, O'Neill H, Zhang Q, Ehlers G, Sokolov AP (2013) Coherent neutron scattering and collective dynamics in the protein. *GFP Biophys J* 105(9):2182–2187. <https://doi.org/10.1016/j.bpj.2013.09.029>
6. Perticaroli S, Nickels JD, Ehlers G, Sokolov AP (2014) Rigidity, secondary structure, and the universality of the boson peak in proteins. *Biophys J* 106(12):2667–2674. <https://doi.org/10.1016/j.bpj.2014.05.009>
7. Pirovano W, Heringa J (2010) Protein secondary structure prediction. *Methods Mol Biol* 609:327–348. https://doi.org/10.1007/978-1-60327-241-4_19
8. Calligari PA, Kneller GR (2012) ScrewFit: combining localization and description of protein secondary structure. *Acta Crystallogr Sect D* 68(Pt 12):1690–1693. <https://doi.org/10.1107/s0907444912039029>
9. Seeliger D, De Groot BL (2010) Conformational transitions upon ligand binding: Holo-structure prediction from apo conformations. *PLoS Comput Biol* 6(1):e1000634. <https://doi.org/10.1371/journal.pcbi.1000634>
10. Xiao X, Wang P, Chou KC (2009) Predicting protein quaternary structural attribute by hybridizing functional domain composition and pseudo amino acid composition. *J Appl Crystallogr* 42:169–173

11. Bu Z, Callaway DJ (2011) Proteins MOVE! Protein dynamics and long-range allostery in cell signaling. *Protein Structure and Diseases. Adv Protein Chem Struct Biol* 83:163–221. <https://doi.org/10.1016/B978-0-12-381262-9.00005-7>
12. Mittag Tanja, Marsh Joseph, Grishaev Alexander, Orlicky Stephen, Lin Hong, Sicheri Frank, Tyers Mike, Forman-Kay Julie D (2010) Structure/function implications in a dynamic complex of the intrinsically disordered Sic1 with the Cdc4 subunit of an SCF ubiquitin ligase. *Structure* 18(4):494–506. <https://doi.org/10.1016/j.str.2010.01.020>
13. Rokde CN, Kshirsagar M (2013) Bioinformatics: protein structure prediction. In: 2013 fourth international conference on computing, communications and networking technologies (ICCCNT), Tiruchengode, pp. 1–5. <https://doi.org/10.1109/icccnt.2013.6726753>
14. Zhai Y, Yang B, Wang L, An B (2009) New trend of protein secondary structure prediction. In: 2009 international symposium on intelligent ubiquitous computing and education, Chengdu, pp 121–124. <https://doi.org/10.1109/iuce.2009.9>
15. Rost B, Sander C (1999) Third generation prediction of secondary structure. In: Protein structure prediction: methods and protocols. Humana Press, New Jersey, USA
16. Kaczanowski S, Zielenkiewicz P (2010) Why similar protein sequences encode similar three-dimensional structures? *Theoret Chem Acc* 125(3–6):643–650. <https://doi.org/10.1007/s00214-009-0656-3>
17. https://en.wikipedia.org/wiki/List_of_protein_structure_prediction_software#Ab_initio_structure_prediction. Accessed on 23rd June 2019
18. Peng Jian, Xu Jinbo (2011) RaptorX: exploiting structure information for protein alignment by statistical inference. *Proteins* 79(Suppl 10):161–171. <https://doi.org/10.1002/prot.23175>
19. Peng Jian, Xu Jinbo (2010) Low-homology protein threading. *Bioinformatics* 26(12):i294–i300. <https://doi.org/10.1093/bioinformatics/btq192>
20. Ma Jianzhu, Wang Sheng, Xu Jinbo (2012) A conditional neural fields model for protein threading. *Bioinformatics* 28(12):i59–i66. <https://doi.org/10.1093/bioinformatics/bts213>
21. Wu S, Zhang Y (2007) LOMETS: a local meta-threading-server for protein structure prediction. *Nucleic Acids Res* 35(10):3375–3382
22. Skolnick J, Kihara D, Zhang Y (2004) Development and large scale benchmark testing of the PROSPECTOR 3.0 threading algorithm. *Protein* 56:502–518
23. Bryant SH, Lawrence CE (1993) An empirical energy function for threading protein sequence through the folding motif. *Proteins* 16(1):92–112
24. Lee, Wu S, Zhang Y (2009) Ab initio protein structure prediction. In: Rigden DJ (ed) From protein structure to function with bioinformatics, pp 3–25. Springer, Netherlands
25. Xu D, Zhang Y (2012) Ab initio protein structure assembly using continuous structure fragments and optimized knowledge-based force field. *Proteins* 80(7):1715–1735
26. Xu D, Zhang Y (2013) Toward optimal fragment generations for ab initio protein structure assembly. *Proteins* 81(2):229–239
27. Thomas PD, Dill KA (1996) Statistical potentials extracted from protein structures: how accurate are they? *J Mol Biol* 257(2):457–469
28. Taylor WR, Bartlett GJ, Chelliah V et al (2008) Prediction of protein structure from ideal forms. *Proteins* 70(4):1610–1619
29. Pedersen JT, Moult J (1997) Ab initio protein folding simulations with genetic algorithms: simulations on the complete sequence of small proteins. *Proteins* 29:179–184
30. Melo F, Sanchez R, Sali A (2002) Statistical potentials for fold assessment. *Protein Sci* 11(2):430–448
31. Oldziej S, Czaplewski C, Liwo A et al (2005) Physics-based protein-structure prediction using a hierarchical protocol based on the UNRES force field: assessment in two blind tests. *Proc Natl Acad Sci USA* 102(21):7547–7552
32. Lindorff-Larsen K, Maragakis P, Piana S et al (2012) Systematic validation of protein force fields against experimental data. *PLoS ONE* 7(2):e32131
33. Freddolino PL, Harrison CB, Liu Y et al (2010) Challenges in protein folding simulations: timescale, representation, and analysis. *Nat Phys* 6(10):751–758

34. Zhang Y, Kihara D, Skolnick J (2002) Local energy landscape flattening: parallel hyperbolic monte carlo sampling of protein folding. *Proteins-Struct Funct Genet* 48(2):192–201
35. Klepeis JL, Wei Y, Hecht MH et al (2005) Ab initio prediction of the three-dimensional structure of a de novo designed protein: a double-blind case study. *Proteins* 58(3):560–570
36. Kryshtafovych A, Barbato A, Monastyrskyy B et al (2015) Methods of model accuracy estimation can help selecting the best models from decoy sets: assessment of model accuracy estimations in CASP11. *Proteins* 84:349–369
37. Jayachandran G et al (2006) Using massively parallel simulation and Markovian models to study protein folding: Examining the dynamics of the villin headpiece. Published online
38. Kmiecik S, Gront D, Kolinski M, Wieteska L, Dawid AE, Kolinski A (2016-06-22) Coarse-grained protein models and their applications. *Chem Rev* 116(14):7898–936. <https://doi.org/10.1021/acs.chemrev.6b00163>
39. Roy A, Kucukural A, Zhang Y (2010) I-TASSER: a unified platform for automated protein structure and function prediction. *Nat Protoc* 5(4):725–738
40. Fujitsuka Y, Chikenji G, Takada S (2006) SimFold energy function for de novo protein structure prediction: consensus with Rosetta. *Proteins* 62(2):381–398
41. Moult J et al (2007) Critical assessment of methods of protein structure prediction—Round VII. *Proteins* 69(Suppl 8):3–9. <https://doi.org/10.1002/prot.21767>
42. Zhang Y, Skolnick J (2005) The protein structure prediction problem could be solved using the current PDB library. *Proc Natl Acad Sci USA* 102(4):1029–1034. <https://doi.org/10.1073/pnas.0407152101>
43. Qian B et al (2007) High-resolution structure prediction and the crystallographic phase problem. *Nature* 450(7167):259–264. <https://doi.org/10.1038/nature06249>
44. Tress M et al (2009) Target domain definition and classification in CASP8. *Proteins* 77(Suppl 9):10–17. <https://doi.org/10.1002/prot.22497>
45. Kryshtafovych A, Monastyrskyy B, Fidelis K (2014) CASP prediction center infrastructure and evaluation measures in CASP10 and CASP ROLL. *Proteins Struct Funct Bioinform* 82(Suppl 2):7–13. <https://doi.org/10.1002/prot.24399>
46. Kryshtafovych A et al (2007) Progress from CASP6 to CASP7. *Proteins: Struct Funct Bioinf* 69(Suppl 8):194–207. <https://doi.org/10.1002/prot.21769>

Computational Methods for Inference of Gene Regulatory Networks from Gene Expression Data



Nimrita Koul and Sunilkumar S. Manvi

1 Introduction

When the genes in a cell express themselves, it leads to synthesis of amino acids for protein synthesis. A protein is the building block of a body and controls the proper functioning of various tissues, organs and organ systems of the body. It is therefore clear that expression of genes controls the health and homeostasis of an organism. For the last few years, the advancement of technologies like DNA microarray, next-generation sequencing, etc. has made available to researchers a huge volume of various kinds of biological data about working of a human cell. Some of this data is available in public repositories over the Internet. Many researchers are using computational methods to derive meaningful information about health of the cell and the organism from this data. These techniques help scientists to integrate data from various sources, look at it from various angles and derive useful insights from it about the diagnosis of the patient.

Gene expression involves works in two steps—the transcription step, in which messenger RNA (mRNA), transfer RNA (tRNA) and ribosomal RNA (rRNA) [1] are produced by the action of enzyme RNA. This mRNA decides the sequence of amino acids in a protein. The second step is translation, and in this step, mature mRNA acts as a template to put together various amino acids in correct sequence to generate a polypeptide inside the ribosomes of a cell. These ribosomes consist of proteins and rRNA. Figure 1 shows the stages of transcription and translation.

Gene expression is regulated by one or many of these ways—regulating the rate of translation or transcription and regulating processing and stability of RNA. Figure 2 shows the process of gene regulation in a cell in detail.

N. Koul (✉) · S. S. Manvi

School of Computing & Information Technology, REVA University, Bangalore, India
e-mail: nimritakoul@revau.edu.in

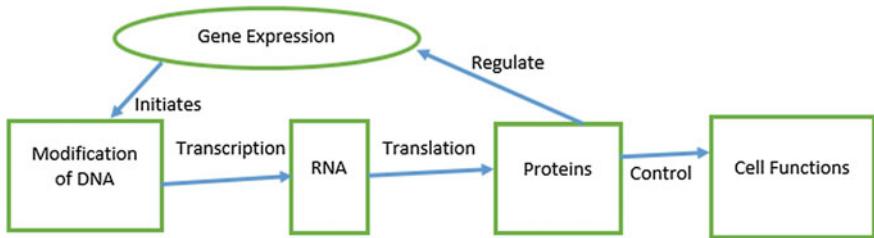


Fig. 1 Stages of translation and transcription in gene expression

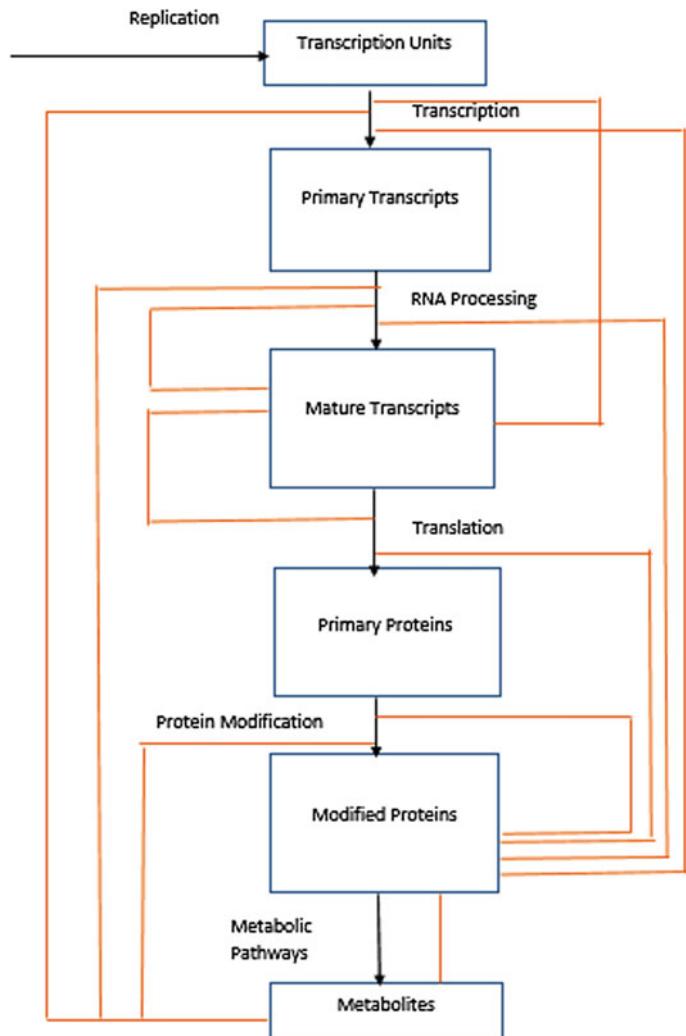


Fig. 2 Sub-stages and transcripts involved in gene expression

2 Gene Regulatory Networks

A gene regulatory network is the representation of the interactions among a set of molecular components which are produced by expression of a gene. These components interact not only among themselves but also with other chemicals in a cell. These interactions in turn regulate the expression of many genes inside a cell and thus influence morphogenesis [2]. The regulators are the DNA, RNA, proteins or a combination of all three. The gene regulatory interactions can be via RNA or protein. Structural proteins or enzyme proteins synthesized from mRNA are responsible for certain structural properties of cell or certain metabolic processes within a cell. Certain proteins, called as transcription factors, have the predefined purpose of activating other genes by binding to their promoter region or inhibiting certain genes. Thus, gene regulatory networks can have a cascading effect. A product of one gene expressing itself may turn on other genes in neighbouring cells. Figure 3 shows an example gene regulatory network in hybrid rice.

The nodes in a gene regulatory network represent any one of the mRNAs, proteins, genes or their complexes. Edges indicate the chemical reactions which lead to activation or inhibition of other genes. The nodes can also have a regulatory effect on their own expression; therefore, the network can have cyclic self-feedback loops. The biochemical reactions inside the cells which lead to regulation of expression of original gene or other genes are graphically modelled in form of a GRN. There are two ways to infer these regulatory networks—clinical studies and computational modelling. Computational modelling or inference of gene regulatory networks has been possible by application of machine learning, [3] data analysis and inference

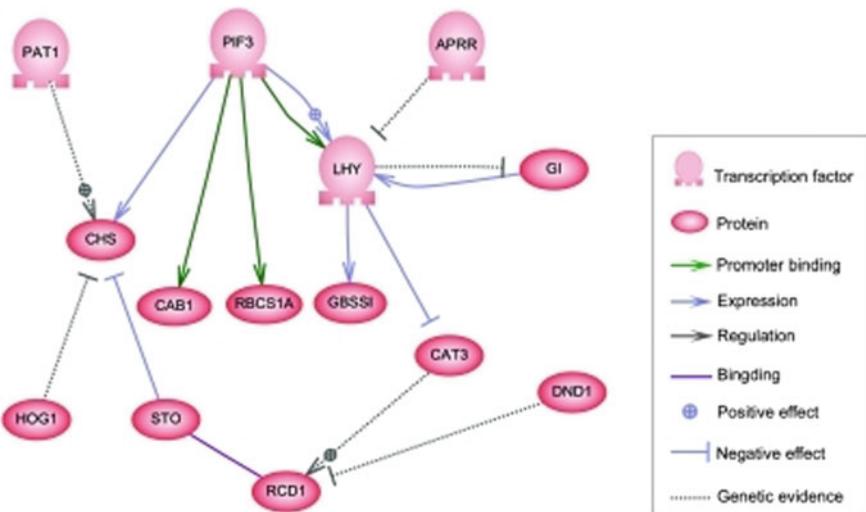


Fig. 3 Example gene regulatory network in hybrid rice. This figure is from https://commons.wikimedia.org/wiki/File:DG_Network_in_Hybrid_Rice.png

techniques on biological data sets like gene expression data. In simplified model of a GRN, genes are considered as nodes, transcription factors are the inputs to nodes and output is the level of expression of this gene. The transcription factors which are inputs are the outputs of a previous step in gene expression. The output can be calculated by employing various modelling techniques. For example, using Boolean networks, the operators like AND, OR and NOT are applied on inputs which simulate the effect that presence of one or more transcription factors has on a gene. Computational or mathematical modelling of gene regulatory networks by analysis of gene expression data or histone modification ChIP-seq data using techniques like ordinary differential equations, Bayesian networks, Gaussian networks artificial neural networks [4] not only helps us in understanding the influential transcripts and their pathways but also we can use the GRNs to predict cell health.

3 Computational Approaches for Construction of Gene Regulatory Networks

Inference of a GRN means reconstructing it from the molecular biological data obtained from experiments, and modelling of a GRN involves building a model for simulation of the changes in gene expression levels in a cell over time. The research literature classifies the methods for GRN inference based on the technique or the property used to find linkages among the transcripts. These methods are classified as network-based methods, regression-based methods, probability-based methods, kernel-based methods and artificial neural network-based methods [5]. The model-based methods for GRN construction include ordinary differential equations, Boolean networks, Bayesian networks, dynamic Bayesian networks and multiple linear regression, supervised learning and neural networks. Model-based methods are computationally very expensive and require parameter tuning. However, by decomposition of large network into “n” sub-networks, the problem can be reduced to finding regulatory networks for single gene at a time. The methods using information theory or correlation of expression values are simplest approaches to construct GRNs. The correlation of values is also called as co-expression. But they do not reveal the direction of regulation.

Boolean networks assume genes to have only two states—ON and OFF corresponding to expressing or not expressing. This state of a gene is assumed to be a function of states of its regulator genes. Statistical clustering-based methods can also be used to identify co-regulated genes.

Bayesian methods being computationally expensive are unsuitable for large scale networks, but since they use probability theory, they are able to predict regulatory relations in noise prone or incomplete biological data. However, modified or hybrid Bayesian network methods have been shown to work reasonably well on large-size networks. Under the category of information theory-based methods, assume the correlation in gene expression as co-regulation and some inherent property of data

like mutual information or entropy is exploited to find co-regulated or dependent genes without telling the direction of regulation. Conditional mutual information can be used to infer indirect regulation. Regression-based methods including neural networks and supervised learning methods [6] like random forests, linear regression, multiple regression are commonly used methods for GRN inference.

Any GRN inference method has to satisfy following criteria:

- i. From input data, it should be able to figure out regulatory relationships.
- ii. It should be able to scale up to find regulatory relations from entire genome of an organism.

3.1 Ordinary Differential Equations

A differential equation is one consisting of derivatives [7]. These equations model the rate of change quantity with respect to other. There are two types of differential equations:

- i. Ordinary differential equations (ODE)
- ii. Partial differential equations (PDE).

In an ODE, the derivatives are calculated with respect to only one independent variable. For example:

$$\frac{d^2y}{dx^2} + 2\frac{dy}{dx} + y = 0, \quad \frac{dy}{dx}(0) = 2, \quad y(0) = 4, \quad (1)$$

$$\frac{d^3y}{dx^3} + 3\frac{d^2y}{dx^2} + 5\frac{dy}{dx} + y = \sin x, \quad \frac{d^2y}{dx^2}(0) = 12, \quad \frac{dy}{dx}(0) = 2, \quad (2)$$

$$y(0) = 4$$

Order of an ODE is equal to the highest order derivative in it, and degree of an ODE is equal to the power of the highest order derivative.

For example, the ODE 3

$$x^3 \frac{d^3y}{dx^3} + x^2 \frac{d^2y}{dx^2} + x \frac{dy}{dx} + xy = e^x \quad (3)$$

has order 3 and degree 1.

And Eq. 4

$$\left(\frac{dy}{dx} + 1\right)^2 + x^2 \frac{dy}{dx} = \sin x \quad (4)$$

has order 1, the order of highest derivative and a degree of 2, the power of highest derivative term. Derivative of a variable x with respect to another variable t is the rate of change of x with respect to t . It is represented as $\frac{dx}{dt}$.

In case of GRN modelling, x is the concentration of transcript x in a cell, and t is the time. Therefore, this derivative is the rate of change in concentration of x over time. Here, x is dependent and t is independent variable. This rate of change can be positive, i.e. rate of increase, or negative, i.e. rate of decrease. Assuming that the rate of increase of x to be proportional to value of x , we can write that $\frac{dx}{dt} = kx$, k being a non-zero, positive constant.

The equation

$$\frac{dx}{dt} = kx \quad (5)$$

is known as an ordinary differential equation consisting of first-order derivative of x with respect to t . When $t = 0$, the value of x is x_0 , the initial condition. If $x = 0$, then the derivative is 0, and this represents a condition of no change in the expression of gene or transcript represented by variable x . This is the trivial solution to above equation. In case x is not equal to 0, we need to solve the differential equation to obtain value of x at next t or a future time value. In general, solving a differential equation means that knowing value of k and x_0 , we can calculate the value of x at each succeeding t . We need to come up with a function, $x(t)$, whose derivative is kx .

Ordinary differential equation can be numerically solved using Euler's method as follows:

For an initial state x_0 and a small discrete Δt , we calculate $(t_k + 1, x_k + 1)$ from the preceding point (t_k, x_k) as follows:

Calculate slope of $f(t_k, x_k)$

Calculate the next point $(t_k + 1, x_k + 1)$ as follows:

$$t_k + 1 = t_k + \Delta t \quad (6)$$

$$x_k + 1 = x_k + f(t_k, x_k) \Delta t \quad (7)$$

In order to model regulatory relations between two genes using ODE, let us assume that $m1$ and $m2$ are the levels of concentration of messenger RNA1 and messenger RNA2, respectively, let $k1$ and $k2$ be two constants indicating the rate of production of messenger RNAs and let $y1$ and $y2$ be two constants indicating rate of degradation of mRNAs. The values of $k1$, $k2$, $y1$ and $y2$ are all constant. The mutual inhibitory regulation between gene 1 and gene 2 can be modelled as with following equations:

$$m1 = k1 f(m2) - y1 m1 \quad (8)$$

$$m2 = k2 f(m1) - y2 m2 \quad (9)$$

where

$$f(m) = \theta^n / \theta^n + m^n, \theta > 0 \quad (10)$$

The rate of change of concentration of mRNA of a gene is

$$\frac{dm}{dt} = (\text{regulator transcription value}) - \text{degradation value}$$

The rate of change of other products of genes is

$$\frac{dp}{dt} = (\text{translation value} + \text{diffusion value}) - \text{degradation value}$$

The classical method to solve a general ordinary differential equation of linear order and constant coefficients is as follows:

$$\frac{d^n y}{dx^n} + k_n \frac{d^{n-1} y}{dx^{n-1}} + \cdots + k_3 \frac{d^2 y}{dx^2} + k_2 \frac{dy}{dx} + k_1 y = F(x) \quad (11)$$

Its general solution y has two components, a homogeneous y_H and a particular part y_P .

$$y = y_H + y_P \quad (12)$$

y_H when substituted in the left-hand side of the equation gives zero so is solution of the equation

$$\frac{d^n y}{dx^n} + k_n \frac{d^{n-1} y}{dx^{n-1}} + \cdots + k_3 \frac{d^2 y}{dx^2} + k_2 \frac{dy}{dx} + k_1 y = 0 \quad (13)$$

The above equation can be written as

$$D^n y + k_n D^{n-1} y + \cdots + k_2 D y + k_1 y = 0 \quad (14)$$

$$(D^n + k_n D^{n-1} + \cdots + k_2 D + k_1) y = 0 \quad (15)$$

where

$$D^n = \frac{d^n}{dx^n}$$

$$D^{n-1} = \frac{d^{n-1}}{dx^{n-1}}$$

$(D - r_1), (D - r_2), \dots, (D - r_n)$ are the factors of the differential equation $D^n + k_n D^{n-1} + \dots + k_2 D + k_1 = 0$.

The solution to above equation involves finding its roots. The roots belong to three categories:

- i. Real and distinct
- ii. Real and indistinct/identical
- iii. Complex.

In case of real and distinct roots, the solution is

$$(D - r_1)y = 0 \quad (16)$$

$$\frac{dy}{dx} = r_1 y \text{ or } \frac{dy}{y} = r_1 dx$$

Integrating both sides

$$\ln y = r_1 x + c$$

$$y = ce^{r_1 x} \quad (17)$$

There are n different solutions corresponding to n different factors given by

$$C_n e^{r_n x}, C_{n-1} e^{r_{n-1} x}, \dots, C_2 e^{r_2 x}, C_1 e^{r_1 x}$$

where $r_n, r_{n-1}, \dots, r_2, r_1$ are the roots and $C_n, C_{n-1}, \dots, C_2, C_1$ are constants and

$$y_H = C_1 e^{r_1 x} + C_2 e^{r_2 x} + \dots + C_{n-1} e^{r_{n-1} x} + C_n e^{r_n x} \quad (18)$$

If two roots are identical, i.e. $r_1 = r_2$, then

$$(D - r_n)(D - r_{n-1}) \dots (D - r_1)(D - r_1)y = 0$$

$$(D - r_1)(D - r_1)y = 0$$

If

$$(D - r_1)y = z$$

then

$$\begin{aligned} (D - r_1)z &= 0 \\ z &= C_2 e^{r_1 x} \end{aligned} \quad (19)$$

The solution is

$$y_H = (C_1 + C_2x)e^{r_1x} + C_3e^{r_3x} + \cdots + C_n e^{r_n x} \quad (20)$$

For m identical roots, the solution is

$$y_H = (C_1 + C_2x + C_3x^2 + \cdots + C_mx^{m-1})e^{r_m x} + C_{m+1}e^{r_{m+1}x} + \cdots + C_n e^{r_n x}$$

If the roots are complex, say $r_1 = \alpha + i\beta$ and $r_2 = \alpha - i\beta$,

$$y_H = C_1 e^{(\alpha+i\beta)x} + C_2 e^{(\alpha-i\beta)x} + C_3 e^{r_3 x} + \cdots + C_n e^{r_n x}$$

Therefore,

$$\begin{aligned} y_H &= C_1 e^{\alpha x} (\cos \beta x + i \sin \beta x) + C_2 e^{\alpha x} (\cos \beta x - i \sin \beta x) \\ &\quad + C_3 e^{r_3 x} + \cdots + C_n e^{r_n x} \\ &= e^{\alpha x} (A \cos \beta x + B \sin \beta x) + C_3 e^{r_3 x} + \cdots + C_n e^{r_n x} \end{aligned} \quad (21)$$

where $A = C_1 + C_2$ and $B = i(C_1 - C_2)$

Particular part of solution y_P is the given by this equation

$$(D^n + k_n D^{n-1} + k_{n-1} D^{n-2} + \cdots + k_1) y_P = X \quad (22)$$

3.2 Neural Networks Method

3.2.1 Recurrent Neural Networks

A recurrent neural network (RNN) is a type of neural network in which the nodes are connected in the form of a directed cyclic or acyclic graph. The graph directions follow a temporal sequence. While an acyclic graph can be unfolded to see the involved states, a cyclic graph cannot be unfolded. Certain RNNs also have a gated memory unit, and such RNNs are known as long short-term memory (LSTM) networks [8]. The nodes at each layer of an RNN are directionally connected to each node in next layer. Activation function is non-linear and weights are real valued. The input values, in the form of input vectors, are fed into input nodes one by one, and a hidden node computes its activation using a non-linear activation function on weighted sum of activations of all its input units and compares it with target activations available at some of the output nodes.

Since gene regulation is a temporal sequence of interactions, involving more than one gene influencing each other in time, we can model the expression regulation using RNNs [9]. A node in RNN models a gene; the edge weights represent regulatory

influence between the gene at source of edge and the destination of the edge. Value of weights of edges in a particular layer in this RNN will represent the value of expression of the gene at the source of the edge at one point in time t_i .

At time $i + 1 = t_i + dt$, the expression level of a gene is determined by expression level of all genes which have an incoming edge to this gene at t_i .

$$\text{i.e. Expression of Gene } i = \sum_{j=1}^n w_{ij} * x_j + Bi$$

Bi is the delay parameter, higher the Bi less is the influence of weight w_{ij} on expression of gene i . Aim of training the RNN is to find optimal values of weights and the delay parameter so as to minimize the mean square error between actual time series data and the regenerated data. This error is given by

Mean Square Error

$$= \frac{1}{\text{Number of Genes } N * \text{Number of Time Points } T} \sum_{i=1}^N \sum_{t=1}^T (x_i(t) - \tilde{x}_i(t))^2$$

A generalized algorithm for training this RNN can be defined as below:

1. Use some domain knowledge or heuristic to select a subset of genes from the genome to begin the process. These will be inputs to the input layer of our RNN [10].
2. Initialize or update the parameter values—weights, delay constants and biases.
3. Compute the estimated value of gene expression of the input genes based on input values of genes and parameters by applying the RNN model. Calculate the estimated gene expression time series based on the RNN model, and evaluate the optimization fitness function for each particle.
4. Optimize the fitness function, i.e. mean square error function, for each node in the current layer of RNN and update the parameter values accordingly.
5. Repeat from step 2 till the optimal value of parameters has been found or the number of iterations has reached a maximum.

3.3 Boolean Network-Based Methods

A Boolean network is a directed graph $G(V, E)$ where V is a set of nodes and E is a set of edges. The nodes are genes and the edges are interactions between the nodes. Boolean variables can assume one of the two values—True or False corresponding to two possible states of a gene—“Turned On” or “Turned Off”. The transitions in the states of genes are based on Boolean logic and are determined by the states of other genes which are connected to it in the Boolean network.

The expression value of a node v at time t_i is given by $v_i(t_i)$ at time t_{i+1} , the expression of the node is given by $v_i + 1(t_i + 1)$. The relation between $v_i + 1(t_i + 1)$ and $v_i(t_i)$ is given by:

$$v_i(t_i + 1) = f_i(v_i 1(t), v_i 2(t), v_i 3(t), \dots, v_i k(t))$$

where f is a Boolean function of the form

$$f_i: \{0, 1\}^{k^i} \rightarrow \{0, 1\}$$

Assuming k regulatory genes, the possible number of Boolean functions is 2^{2^k} for f_i .

The aim is to infer update functions from time series-based gene expression datasets and the interactions [11]. The correctness of these interactions can be measured by comparison between the actual and generated trajectory. One possible means of measuring similarity between Boolean trajectories is gene consistency, $\text{Con}(V1, V2)$ which is given as follows:

$$\text{Con}(V1, V2) = \frac{\sum_{t=2}^T I(v(t) = \dot{v}(t))}{T - 1}$$

T is total time steps, and I is Boolean function returning 1 if the condition is True else 0. Figure 4 shows a gene regulatory network consisting of three genes with regulatory interaction among them.

Tables 1 and 2 give the states of the above three genes at times T0 and T1.

At T1, the states are:

The rules inferred by the Boolean network about the regulatory interactions of these three genes are:

Gene 1 = $-$ Gene 1, i.e. Gene 1 has an inhibitory effect on itself

Fig. 4 A sample gene regulatory network

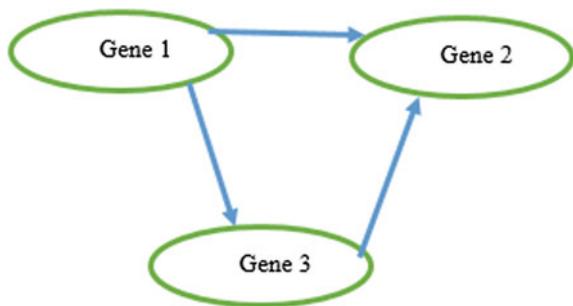


Table 1 State of three genes at T0

Gene 1	Gene 2	Gene 3
Turned off	Turned off	Turned off
Turned on	Turned off	Turned off
Turned off	Turned on	Turned on
Turned on	Turned off	Turned off

Table 2 State of same three genes at T1

Gene 1	Gene 2	Gene 3
Turned on	Turned off	Turned off
Turned off	Turned on	Turned on
Turned on	Turned off	Turned on
Turned off	Turned on	Turned on

Gene 2 = Gene 1—Gene 3, i.e. Gene 2's expression is OR operation between expressions of Gene 1 and Gene 3

Gene 3 = Gene 1—Gene 3, i.e. Gene 3's expression is OR operation between expressions of Gene 1 and Gene 3.

Boolean networks have following subtypes—random Boolean networks, asynchronous Boolean networks, [12] temporal Boolean networks and probabilistic Boolean networks.

A general algorithm for inference of regulatory relations using Boolean networks is as follows:

- i. Input the appropriate gene expression data set
- ii. Identify the candidate genes to initialize the Boolean network using any of the approaches like information theory-based analysis, genetic algorithms, PSO, etc.
- iii. Determine initial node connections
- iv. Begin iterations of the algorithm for values of K from 1 to n. That is, determine relations between two genes
- v. Increment K by 1 and identify relation between 3 genes and so on.

3.4 Bayesian Network-Based Methods

A Bayesian network or a belief network is a graphical model to represent multivariate probability distributions among the variables in a system under study. Nodes of the directed acyclic graph show the variables and the edges represent dependencies.

For a variable X_i which is one of the nodes of the graph, the distribution of its probabilities is represented as:

$$P(X_1, X_2, X_3, \dots, X_N) = \prod_{i=1}^N P(X_i | \prod X_i)$$

Here, $\prod X_i$ is a set of parent vertices of i .

Each node of the graph, i.e. each variable X_i has a probability table:

$$P(X_i | \prod X_i)$$

Directed acyclic graphs can easily express the probability distributions of this nature.

Therefore, we can formally describe a Bayesian network as a directed graph $G(V, E)$ with two components

- i. A set of nodes X_i where X_i belongs to V .
- ii. A set of edges E containing a table of conditional probability distributions for each node linking parent node to child node.

Consider the example where we wish to diagnose if a person is suffering from pneumonia or a common cold, represented by symbols P and C , respectively. So we have a set of symptoms to consider, e.g. flowing nose, headache and hot flashes. Let us denote these symptoms by the symbols N , H and F , respectively. Also let us take into consideration the information whether the patient has had pneumonia within last one year, represented by variable R .

Then, the Bayesian network for this diagnosis can be drawn like shown in Fig. 5.

Applying Bayesian Networks for Reconstruction of Gene Regulatory Networks

In order to search a network structure in the given set of genes whose expression values are given as input, we first need to use some information theoretic measure like mutual information, conditional mutual information, Kullback–Leibler (KL) divergence, to identify correlation and non-linear dependencies among pairs of genes or a group of genes [13, 14].

A generalized algorithm for using Bayesian network to infer GRNS is as follows:

- i. Identify the most relevant genes for the problem being investigated by following a feature selection method.

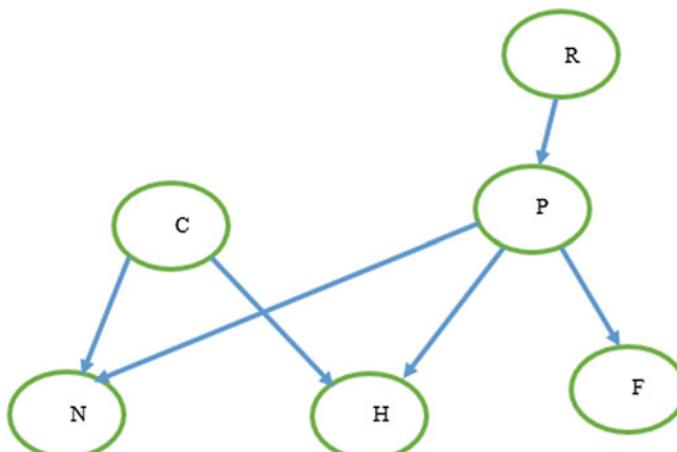


Fig. 5 A simplified Bayesian network to show probability of a cold being pneumonia or common cold

- ii. Calculate conditional dependencies among the genes in this selected group using Bayes' rule and draw the initial network.
- iii. From the calculated dependency values, treat the ones above a threshold value to be valid and discard the ones below threshold and reconstruct the network with only valid dependencies.

Many researchers have used supervised and unsupervised machine learning methods to calculate the threshold and reconstruct the network.

4 Conclusion

In this chapter, we looked at prominent computational techniques used for inference of gene regulatory networks. With the advances in artificial intelligence, machine learning and high throughput sequencing techniques and computing power of the hardware, it has become easier to apply data analysis techniques to gather deeper insights from biological data. Use of computational techniques to infer GRNS has led to discovery of many interlinked biological phenomena. Research into diseases like cancer has been benefitted by use of such techniques; identification of gene regulatory networks can help in early diagnosis or drug discovery for diseases. We described four major computational techniques in this chapter, viz. ordinary differential equations, recurrent neural networks, Boolean networks and Bayesian networks. The ordinary differential equation method is highly mathematical but straight forward system to model the gene regulatory networks. Their drawback is the complexity in understanding them. The Boolean network method is simple but cannot handle large networks. The Bayesian Network method is a balance of above two approaches which described dependencies between genes by way of their joint probabilities. Recurrent neural network method takes into account the temporal expression values of the genes at different layers to model the relations among them.

References

1. Godini R, Karami K, Fallahi H (2019) Genome imprinting in stem cells: a mini-review. *Gene Expr Patterns* 34:119063. ISSN 1567-133X. <https://doi.org/10.1016/j.gep.2019.119063>
2. Dewey GT, Galas DJ (2010) Gene regulatory networks. In: Madame curie bioscience database (Landes Bioscience, Austin, TX). <https://www.ncbi.nlm.nih.gov/books/NBK5974/>
3. Chan TE, Stumpf MPH, Babtie AC (2017) Gene regulatory network inference from single-cell data using multivariate information measures. *Cell Syst* 5:251–267.e3. <https://doi.org/10.1016/j.cels.2017.08.014>
4. Huang J, Shimizu H, Shioya S (2003) Clustering gene expression pattern and extracting relationship in gene network based on artificial neural networks. *J Biosci Bioeng* 96(5):421–428
5. Camacho DM, Collins KM, Powers RK, Costello JC, Collins JJ (2018) Next-generation machine learning for biological networks. *Cell* 173:1581–1592. <https://doi.org/10.1016/j.cell.2018.05.015>

6. Ni Y, Aghamirzaie D, Elmarakeby H, Collakova E, Li S, Grene R et al (2016) A machine learning approach to predict gene regulatory networks in seed development in *Arabidopsis*. *Front Plant Sci* 7:1936. <https://doi.org/10.3389/fpls.2016.01936>
7. Zhang Q, Yu Y, Zhang J, Liang H (2018) Using single-index ODEs to study dynamic gene regulatory network. *PLOS One* 13(2):e0192833. <https://doi.org/10.1371/journal.pone.0192833>
8. Omranian N, Eloundou-Mbebi JM, Mueller-Roeber B, Nikолоски Z (2016) Gene regulatory network inference using fused LASSO on multiple data sets. *Sci Rep* 6:20533. <https://doi.org/10.1038/srep20533>
9. Mandal S, Saha G, Pal RK (2017) Recurrent neural network-based modeling of gene regulatory network using elephant swarm water search algorithm. *J Bioinform Comput Biol* 15(4), 1750016. <https://doi.org/10.1142/S0219720017500160>
10. Greenfield A, Hafemeister C, Bonneau R (2013) Robust data-driven incorporation of prior knowledge into the inference of dynamic regulatory networks. *Bioinformatics* 29:1060–1067. <https://doi.org/10.1093/bioinformatics/btt099>
11. Barman S, Kwon YK (2018) A Boolean network inference from time-series gene expression data using a genetic algorithm. *Bioinformatics* 34(2018):i927–i933. <https://doi.org/10.1093/bioinformatics/bty584>
12. Huynh-Thu VA, Irrthum A, Wehenkel L, Geurts P (2010) Inferring regulatory networks from expression data using tree-based methods. *PLoS One* 5:e12776. <https://doi.org/10.1371/journal.pone.0012776>
13. Liu F, Zhang SW, Guo WF, Wei ZG, Chen L (2016) Inference of gene regulatory network based on local Bayesian networks. *PLoS Comput Biol* 12:e1005024. <https://doi.org/10.1371/journal.pcbi.1005024>
14. Rogers S, Girolami M (2005) A Bayesian regression approach to the inference of regulatory networks from gene expression data. *Bioinformatics* 21(14):3131–3137

Machine-Learning Algorithms for Feature Selection from Gene Expression Data



Nimrita Koul and Sunilkumar S. Manvi

1 Introduction

What does it mean for a machine to learn? In other words, what is machine learning? Machine learning is the process of making a computer figure out, as closely as possible, the mathematical relationship between a set of inputs and a set of outputs for a problem under consideration, i.e. the machine tries to arrive at the best function $f()$, also known as a model, between input dataset X and output set Y . Input set X may consist of many data points x_1, x_2, \dots, x_n , where each x_i is a value of any data type.

All members x_i of X are known as features of input set, however, it has been seen that not all of x_i equally influence the output, i.e. there is a subset of features in X that are sufficient to arrive at the output Y . Remaining features are irrelevant and can be ignored for the purpose of learning the relationship between X and Y . We identify the subset of relevant features and use only them to establish the relation between X and Y . This reduces the computational complexity of learning the model. Because we drop the irrelevant features, the chances of overfitting in the model are reduced. The process of identifying the smallest and most relevant subset of features that are sufficient to establish a valid relation between inputs and outputs for a problem is known as feature selection.

In the field of bioinformatics and computational biology, medical data, e.g. gene expression data is an indicator of function and state of all genes in an organism's genome and can be used to diagnose a disease like cancer. However, to the problem of diagnosis of cancer, only a few genes in the total genome of thousands of genes are relevant; therefore, it is imperative to try and identify these few genes from the expression of entire genome, and then use the expression values of these few genes to classify a patient as having the disease or not.

N. Koul (✉) · S. S. Manvi

School of Computing & Information Technology, REVA University, Bangalore, India

e-mail: nimritakoul@reva.edu.in

© Springer Nature Singapore Pte Ltd. 2020

151

K. G. Srinivasa et al. (eds.), *Statistical Modelling and Machine Learning Principles for Bioinformatics Techniques, Tools, and Applications, Algorithms for Intelligent Systems*, https://doi.org/10.1007/978-981-15-2445-5_10

2 Gene Expression Data

Gene expression data [1] is the numerical data corresponding to the expression level of genes in the cell of an organism. It is obtained using DNA microarray technology. DNA microarray technology involves the hybridization of an mRNA (a subcellular component) molecule to the DNA template. Amount of mRNA bound to each site on array indicates expression level of genes in the sample. This data is known as the gene expression profile of the cell. This data can reveal presence of diseases and even the possibility of occurrence of certain diseases at the transcriptome level. A study of these profiles can also help in development of drugs to target the disease-causing genes and their action pathways. In addition to DNA microarray technology, we now have an advanced technology called next-generation sequencing or high throughput sequencing which is a process of determining the sequence of nucleotides in a particular portion of DNA or RNA. It can reveal any abnormal or out of sequence nucleotides and hence indicate an abnormal bodily condition or a disease. In this chapter, we focus on use of machine learning for feature selection from gene expression datasets. A prominent characteristic of gene expression datasets is very high dimensionality, of the order of thousands of genes present in the genome of an organism, whereas the sample size is very small. The noise and redundancy further make preprocessing of the data mandatory. To control the curse because of high dimensionality, feature selection is performed on the gene expression data to identify the smallest subset of genes which are accurate and sufficient markers of a disease or a condition that is the subject of study. Identification of such genes not only reduced the computational complexity of entire machine-learning effort to classify or cluster the profiles but also helps in discovery of accurate drugs and their targets in order to treat or cure certain diseases like cancer.

The machine-learning approaches are used to identify differentially expressed genes which are markers or a disease and also to eliminate the redundant genes or to identify the most relevant genes,

Prominent Applications of Gene Expression Data

- i. Identification of new genes in the genome, the genes whose functionality may have been yet unknown can be un-covered during the analysis of gene expression data.
- ii. As discussed, analysis of gene expression data can reveal presence of a disease like cancer.
- iii. Analysis of genes in a diseased and a control cell can help understand the molecular pathways and mechanisms. This knowledge can be used for synthesis of targeted drugs which can control a harmful over activity of a gene, or can promote under active genes to work better for better health of the cell.
- iv. Gene expression data also can be used to research and study the effect of various environmental factors, toxins, stress, etc., on gene activity and thereby on health of a cell.

3 Feature Selection

In the field of machine learning, feature selection [2] is the step that involves elimination of attributes or features that are noisy, irrelevant and redundant for classification of data samples within a given dataset. This elimination is often done on basis of a score, rank or weight that represents relevance of the feature. Feature extraction [3] is another technique of dimensionality reduction which involves combination or transformation of basic features into more complex features in higher dimensions using methods like principal component analysis. Both feature selection and feature extractions prepare a reduced set of most relevant features that capture the essential properties of the data and will correctly classify a given dataset. This classification can be done in a supervised as well as unsupervised way. Supervised learning uses the datasets with output class labels present, while unsupervised learning uses the data with no output class labels available. In supervised learning, a model is computed which represents the equation between pattern in input feature values and the output class labels. Input feature values are called training data. These training samples have their actual class labels present in the data. After training, the test samples are input to the model, test samples do not have a class label, the model tried to predict the label for each sample in test data based on the pattern it has learnt from relation between sample values and output labels seen in training data. The ratio of correctly classified samples to total number of see samples is called as the classification accuracy of a classifier. Therefore, the correct selection of effective features is as important for higher classification accuracy as is the method used for classification.

Feature selection, therefore, is a set of operations which when performed on a dataset with n features, returns a subset of k features, such that $k < n$, and these k features are the sufficient and relevant subset of features for the machine-learning problem under consideration to be solved. Literature classifies feature selection methods into three categories—filter methods, wrapper methods and embedded methods.

Feature selection is a well-researched area in machine learning and the improved algorithms based on evolutionary and bioinspired computing continue to be introduced to improve the performance and accuracy of feature selection.

3.1 A Basic Algorithm for Feature Selection

- i. From the set of input features, X , with cardinality n , pick a subset of minimum possible size.
- ii. Evaluate the performance of this subset by determining the relation between the inputs in this subset and the outputs.
- iii. Use cross validation on test set and validation set of inputs to establish the relationship.
- iv. Now pick a subset with cardinality 1 greater than the previous subset and repeat steps ii to iv till we find the subset with best performance.

A feature selection algorithm finds a subset of optimal features on the basis of an evaluation metric based on the choice of the evaluation metric research literature that has divided feature selection methods into three categories

- i. Filter methods
- ii. Wrapper methods
- iii. Embedded methods.

Filter methods are based on the concepts of information theory and employ the inherent properties of data values in the input data to find their relevance to or influence on the specified or desired output values. This class of methods uses statistical operators to evaluate the influence of the input values on output value. These methods have a lesser computational complexity as compared to other two classes of methods. In most cases, a variable relevance score is calculated, and low-scoring variables are removed. Afterward, the ‘relevant’ variable subset is input into the classification algorithm. Information theory measures like mutual information, χ^2 , F-score, entropy, etc., have been used. The filter methods select the relevant features first and then use them to fit the classification algorithm, i.e. the intrinsic properties of the data decide the score of the feature subset without depending on the classification algorithm. These methods constitute a preprocessing step for many wrapper methods. There are four measures used in filter methods—interclass distance, consistency (Chi-Square), association or correlation between feature and output class, mutual information, information gain, etc. A few ways to measure correlation are Pearson’s correlation, Spearman’s correlation, and Kendall’s correlation. Mutual information is the measure of information in a feature that can be inferred from another feature. Chi-square measures the degree of influence that a feature has on the output class. More the influence more is the information content of the feature with respect to its ability to predict the output class.

Wrapper methods [4] evaluate the relevance of selected subset of genes by observing the classification accuracy obtained by using each selected subset against the actual output. There are two parts to this approach of feature selection—a method to identify relevant subset of genes and a validating classifier that evaluates the classification accuracy obtained using the selected subset of features. Thus, a wrapper method optimizes a predictor during feature selection process, while filter method of feature selection is independent of predictor. Because the filter methods are simple and incur less computational cost they are more suitable for very high-dimensional datasets, however, since wrapper methods give better accuracy many researchers use hybrid approaches involving the benefits of both the filter and wrapper groups of feature selection methods. These methods thus obtain the subset of relevant genes by training and evaluating the performance of a specific classification algorithm. The search for relevant features (feature selection algorithm) is wrapped around a classification algorithm. This search method can be usual search algorithms or heuristic search algorithms based on the need of the application and the nature of the original dataset. In these methods, a new classifier model is trained for each of the new subsets being considered, accuracy of this classifier decides the rank of the feature

subset. These methods give the best subsets of features at the cost of additional computational complexity due to the need for retraining of classifier for each subset.

These methods can search for the probable subsets of features using one of the three search techniques

- i. Exhaustive search techniques like breadth-first search and depth-first search.
- ii. Heuristic search techniques like forward selection that starts from empty subset and recursively adds to it a feature that optimizes the fitness function or backward selection that starts with full feature set in original data and drops one feature from it at a time while optimizing the fitness function. A bidirectional search uses forward and backward search simultaneously and when both FS and BS arrive at the same subset of features, the search stops. Random feature elimination (RFE), Plus-L Minus-R (LRS) are examples of heuristic search techniques. Other examples include genetic algorithms which treat all possible subsets (chromosomes) of original feature set as starting population of possible solutions to the feature selection problem, a set of operators is applied to this initial population to evaluate their fitness. The subsets with higher fitness act as population for next iteration for the algorithm, the next generation. If no subset qualifies for next generation, new subsets are produced by recombination of subsets in previous generation. Crossover and mutation functions are primarily used to produce next generation of chromosomes till an optimal set is obtained. GA can be combined with other feature selection methods to generate initial population of subsets.
- iii. Random search techniques—In this technique, a subset of features is randomly generated and any of the above techniques can be applied on subsets to arrive at optimal subsets.

In wrapper methods, a validation set is often used to compute validation accuracy using techniques like k -fold cross validation. The wrapper methods are slower than filter methods, have a tendency to overfitting and the results have high variance.

Embedded methods have feature selection integrated into classification algorithm. Classification algorithm is used to evaluate the performance of subsets of selected genes but the classifier in this case is a part of the feature selection algorithm itself and not a separate layer. These methods have a built in classifier and, therefore, they are specific to the learning problem, e.g. support vector machine algorithm with linear kernels, these approaches also use a penalization term for the features to differentiate the features. Lasso with penalty is also an example of embedded approach, and another embedded algorithm is decision tree. In this algorithm, a feature subset is selected in each step of growth of tree by dividing a bigger subset into smaller subsets. For more informative features, i.e. more is the relevance of a feature, more is number of child nodes in its subtree belonging to the same output class. Examples of tree-based algorithms for feature selection re CART, ID3, C4.5, etc.

Thus, we can say that in embedded methods the feature selection forms the part of training of the predictive model like artificial neural network, decision trees, support vector machines, etc. These methods often use backward elimination to remove irrelevant features and regularization terms. Embedded methods are slower, and tend

to over-fit in absence of large amount data. But perform better than filter methods in presence of large amounts of input data.

There also is a technique of joining together wrapper and embedded methods known as Ensemble learning. These combine the benefits of both constituent methods and provide robust feature selection with less variance. Examples are bagging and random forest, gradient tree boost, etc.

4 Fitness Measures of a Feature

A subset of features is selected from among the full feature set of input data by evaluating the fitness of various possible subsets of features. There are various measures to compute this fitness. Fitness of a feature decides its relevance to the objective function. Interclass distance, entropy, information gain, and probability of error are some of the inherent properties [5] used to rank features for their relevance.

Probability of Error

Given the objective of correctly classifying the samples in a gene expression dataset, the goal of feature selection algorithm is to find the genes that minimize the probability of classification error of the samples, i.e. maximize the classification accuracy.

Divergence

Divergence measures the separation or distance between conditional probabilities of various classes in the output. Good features are those for whom the divergence in conditional probabilities is more than a threshold value. A few examples of measures of divergence are

- i. Kullback–Liebler divergence
- ii. Patrick-Fisher divergence
- iii. Matusita divergence
- iv. Chernoff divergence
- v. Bhattacharyya divergence
- vi. Kolmogorov divergence.

Association or Dependence

This measures the extent of association or correlation among features, e.g. correlation coefficient. Interclass distance: We assume different output classes to be widely distant in the sample space. Any features that maximize this space are good features, e.g. Euclidean distance.

Entropy

Let X be variable from the set $S = \{x_1, x_2, \dots, x_n\}$, if $P(X)$ is the probability distribution of X , then entropy of random variable X is written as $H(X)$:

$$H(X) = - \sum_{x \in X} P(x) \log P(x)$$

Mutual Information

In order to measure the relationship between two random variables sampled at a time, we use the concept of mutual information.

For two random variables x and y , if $P(x, y)$ is their joint probability distribution, then their mutual information is given by

$$MI(X, Y) = \sum_{x \in X} \sum_{y \in Y} P(x, y) \log \frac{P(x, y)}{P(x)P(y)}$$

Chi-Square Statistic

In this feature selection method, the gene values are converted into discrete intervals based on entropy and then the chi-square, χ^2 , statistic is calculated between each gene and the output class. Chi-square, χ^2 is a statistic used to evaluate influence on a variables value on output classes.

The formula for calculating χ^2 of a gene is given as

$$\chi^2 = \sum_{i=1}^m \sum_{j=1}^k \frac{(A_{ij} - R_i \cdot C_j)^2}{\frac{R_i C_j}{N}}$$

where, m is the number of discrete intervals in data, k is the number of classes in output, N is total number of samples, R_i is number of samples in i th interval, C_j is number of samples belonging to j th class and A_{ij} is number of samples in i th interval and belonging to j th class. Higher the chi-square value, higher is the rank of the genes.

Information Gain

Information gain (IG) is used to measure the relevance of attribute X_i in class V. It calculates the amount of information inherent in the value of the feature. Information gain is the amount of information gained by evaluating a feature that can help in classification of the sample in correct class.

Features that help in correct classification of samples, have highest information, unrelated features provide no information. Information gain is equivalent to decrease in entropy in the system. Where entropy is a measure of impurity in a dataset.

Information gain from attribute A for class C is given as

$$I(C, A) = H(C) - H(C|A)$$

where $H(C)$ is entropy of the class C given by

$$H(C) = - \sum_{c \in C} p(C) \log p(C) \text{ and}$$

$$H(C|A) = - \sum_{c \in C} p(C|A) \log p(C|A)$$

is the conditional entropy of class C given the attribute A.

Correlation-Based Methods

In these methods, various correlation coefficients [6] are calculated between each gene and the output class. A strong correlation coefficient means higher relevance of the gene, a weak correlation means less relevance, i.e. the feature with weak correlation may or may not be necessary for correct classification of the genes. Non-linear correlation is measured by the entropy which measures amount of un-certainty of a data value. This method ignores the features with low correlation with output or class value as irrelevant features with high correlation among themselves are treated as redundant. A feature is relevant if it can predict class of samples correctly which has not been yet achieved using another smaller subset of features.

$$\text{Predictive Capacity of a subset of } k \text{ features} = \frac{(\text{Number of features in subset i.e. } 'k' * \text{Mean correlation between feature and output class})}{(\sqrt{k + k(k - 1)} * \text{average interfeature correlation})}$$

Some commonly used feature selection algorithms

1. Relief Algorithm

The relief algorithm [7] is a multivariate feature selection algorithm based on k -nearest neighbors. This algorithm is a filter method that calculates a score for each feature used to rank each feature. The score is calculated using the distance between pairs of nearest neighbors of the data value. The score of a feature is reduced if the value difference lies in neighboring data pair of same output class. The feature score is increased if value difference of a feature is in the neighboring instance pair who belongs to separate classes. There are various techniques for selecting the nearest neighbors and assigning weights to instances. Relief algorithm can work in presence of noisy and missing data, work with discrete as well as continuous data and handle multiple class classification, have less computational complexity, can give best features even in presence of interactions among features. There are many variant algorithms based on this approach of relief algorithm.

Algorithm

- i. Let X be a two-class dataset with dimensionality p and number of samples n.
- ii. Scale the input data to lie in the interval [0, 1]. I.e. Convert the input data to binary format if not already in Binary format.
- iii. Initialize the score of p features as zeros
 For i = 1 to m iterations:
 - a. Choose a random sample S from the dataset, take all its features in vector X,
 - b. Find the nearest neighbor of this randomly chosen sample using Euclidean distance as a measure of nearness. Closest neighbor belonging to same class as S is known as ‘near hit’ and a closest neighbor belonging to a different class than S is known as near miss.
 - c. Update the score vector as follows
 - i. Decrease the score of a feature if it differs from that feature in the neighboring sample of belonging to same class as S.
 - ii. Increase the score of a feature if it differs from that feature in neighboring sample belonging to different classes.

New Score of feature i

$$\begin{aligned}
 &= \text{Old score of feature } i - (\text{Value of feature } i \text{ in sample } S \\
 &\quad - \text{value of feature } i \text{ in near hit})^2 \\
 &\quad + (\text{Value of feature } i \text{ in sample } S \\
 &\quad - \text{value of feature } i \text{ in near hit})^2
 \end{aligned}$$

- d. Divide the score vector by m, this is the relevance vector of all ‘p’ Features of which genes with relevance greater than a threshold Value can be selected for further work.

Various variants of this relief algorithm have been proposed in literature.

2. Support Vector Machine with Recursive Feature Elimination

Support Vector Machine with Recursive Feature Elimination (SVM-RFE) [8] is a wrapper method that uses recursive feature elimination to arrive at the optimal subset of most relevant genes. The subset is initialized to contain entire set of original features, and then recursively and randomly drop genes from this set, evaluate the resulting set using SVM classifier and repeats the procedure till the minimal subset is found.

The selected subset is evaluated for fitness by computing the classification accuracy of the SVM classifier when trained using this subset of genes. This approach has proven to be robust to overfitting and noise in gene expression datasets.

Generalized SVM-RFE algorithm:

- i. For an input gene expression, dataset X with n features and p samples, initialize the optimal subset S with all n features of the original dataset X
- ii. Train the SVM classifier using all the features in S
- iii. Compute the rank of genes
- iv. Discard the genes with lowest rank and retain 'r' top-ranked genes
- v. Repeat the steps i to iv with newly obtained gene subset till performance of the classifier continues to improve.
- vi. Return the subset with optimal classifier performance.

Support Vector Machines (SVM) Since gene expression data is highly non-linear, it is almost impossible to separate it into separate classes using linear classification algorithms; therefore, we need the algorithms that can map the input data into alternative dimensional spaces and then try to find the planes that can separate the input data into different classes based on the properties of data in higher dimensions. Support vector machine [9] is one such algorithm. SVM is machine-learning algorithm that computes a hyperplane of maximum margin in a higher-dimensional space such that the values in input dataset can be linearly separable. SVM uses a kernel and a penalty term. There are four basic kernels used with SVM linear kernel, polynomial kernel, radial basis function and sigmoid kernel.

Principal Component Analysis (PCA) PCA [10] is a method for feature engineering which leads to dimensionality reduction of the high-dimensional data. This is an unsupervised method for feature reduction [11] which projects data from a high-dimensional space to a low-dimensional space while retaining the features that count for maximum variance in data. PCA involves computing of mutually orthogonal principal components based on Eigen values of the covariance matrix of input data.

Generalized PCA Algorithm

- i. From the input data, produce the matrix of dimensions $N * d$ where, N is number of samples and d is the number of features in each sample.
- ii. Normalize and standardize each value in the matrix obtained in step 1
- iii. Calculate covariance matrix for this matrix
- iv. Calculate Eigen vectors and Eigen values for above matrix
- v. Principal components are the Eigen vectors with largest Eigen values.
- vi. Return the principal components.

5 Conclusion

In this chapter, we have seen various techniques that are used in machine learning for the feature selection step. It has been observed that filter methods generally work well for feature selection from gene expression data. Properties of data like

mutual information or simple correlation coefficients are sufficiently indicative of the relevance of the features. Although to improve the accuracy of classification of the model and computational complexity of the classification, it has been observed that an ensemble of various techniques gives the best results. For example, an ensemble of Bayesian networks and random forests given high accuracy of classification in relatively short time than purely filter feature selection-based classification. The classification models that take care of non-linear dependencies in the features outperform the ones that take features into consideration individually.

References

1. Lu Y, Han J (2003) Cancer classification using gene expression data. *Inf Syst* (Elsevier) 28(4):243–268
2. Guyon I, Weston J, Barnhill S, Vapnik V (2002) Gene selection for cancer classification using support vector machines. *Mach Learn* 46(1–3):389–422
3. Srinivasa KG, Venugopal KR, Patnaik LM (2006) Feature extraction using fuzzy c-means clustering for data mining systems. *Int J Comput Sci Netw Secur* 6(3A):230–236
4. Kohavi R, John G (1997) Wrappers for feature subset selection. *Artif Intell J* (Special issue on relevance) 97(1–2):273–324
5. Sánchez-Marono N, Alonso-Betanzos A, Tombilla-Sanromán M (2007) Filter methods for feature selection—a comparative study. In: Yin H, Tino P, Corchado E, Byrne W, Yao X (eds) Intelligent data engineering and automated learning—IDEAL 2007. IDEAL 2007. Lecture notes in computer science, vol 4881. Springer, Berlin, Heidelberg
6. Yu L, Liu H (2003) Feature selection for high-dimensional data: a fast correlation based filter solution. In: Proceedings of the twentieth international conference on machine learning, ICML, pp 856–863
7. Robnik-Sikonja M, Kononenko I (2003) Theoretical and empirical analysis of ReliefF and RRelief. *Mach Learn* 53:23–69
8. Lin X, Yang F, Zhou L et al (2012) A support vector machine recursive feature elimination feature selection method based on artificial contrast variables and mutual information. *J Chromatogr B Anal Technol Biomed Life Sci* 10:149–155
9. Arenas-García J, Perez-Cruz F (2003) Multi-class support vector machines: a new approach. In: Proceeding of the IEEE international conference on acoustics, speech, and signal processing (ICASSP '03), vol 2, pp 781–784
10. Song F, Mei D, Li H (2010) Feature selection based on linear discriminant analysis. In: 2010 International conference on intelligent system design and engineering application, Changsha, pp 746–749. <https://doi.org/10.1109/isdea.2010.311>
11. Taveira De Souza J, Carlos De Francisco A, Carla De Macedo D (2019) Dimensionality reduction in gene expression datasets. *IEEE Access* 7:61136–61144

Genomics and Proteomics

Unsupervised Techniques in Genomics



Mrinmoyee Bhattacharya

1 Introduction

Learning is a process to acquire knowledge or upgrade the existing knowledge. Initially, humans and animals had this capability to learn, but now it is also possessed by machines. Machine learning is the systematic study and development of algorithms which improve with experience. For example, in genomics, machine learning is used to recognize the locations of transcription start sites (TSS) in a genome sequence. Machine learning generally helps us to discover new structures which are unknown to humans, e.g., data mining. Machine learning finds the hidden pattern within complex data and tries to make decisions or predict future decisions. Basically, machine learning is divided into two categories:

- Supervised learning
- Unsupervised learning

Supervised Learning: In this type of machine learning technique, we already decide the output, and the function maps the input to the desired output. This learning technique where the desired output is given initially is known as supervised learning. In this case, the desired outputs are known as targets [1].

Unsupervised Learning: In this type of learning technique, we find interpretations from the datasets which contain the input data but do not have any definite output which is labeled. The data from the dataset is processed using various machine learning methods. This method is known as unsupervised learning, and a common algorithm using this technique is clustering analysis which is used to explore hidden data from a group of data.

M. Bhattacharya (✉)
St. Joseph's College, Bangalore, India

1.1 Machine Learning in Bioinformatics [2]

Bioinformatics is a science which is interdisciplinary, and it generally uses biological data which has got a computational approach. Machine learning uses the technique of automatically learning from the dataset. The task of DNA sequence classifier is to learn the functions of the new protein which is used in genomic research. Initially, bioinformatics algorithms are used to explicitly program the problems such as predicting the protein structure which was an extremely difficult task. Deep learning techniques automatically learn all the features of the dataset and then creates an abstract dataset which can be used for future learning. This type of multi-layered approach helps in predicting complex datasets when we have to deal with large amount of data. Machine learning techniques can be used in six main subfields of bioinformatics:

- **Genomics**
- **Proteomics**
- **Microarrays**
- **Systems biology**
- **Evolution**
- **Text mining.**

Genomics: It is a subdivision of biotechnology which relates the methods of genetics and molecular biology for genetically representing and sequencing the sets of DNA. Sometimes it does the full genome for certain organism. In this case, the results are organized into databases so that it can be used for applications in data such as medicine or biology [2].

Proteomics: It is also a subdivision of biotechnology that analyzes the group of proteins which are produced by a type of cell so that it can recognize its structure and function. It examines the effect of protein and how it affects the processes of the cell or the external environment. Protein performs a number of activities inside the cell. Every organism has thousands of distinct proteins and peptides within themselves. The objective of proteomics is to analyze the types of proteomes at different times and to find the difference between them. The complete set of proteins which are produced by a type of cell is called proteome. For example, the cancerous cell protein content is different from the normal healthy cell protein.

The techniques used to determine the size of a protein or protein complex is mass spectrometry. With the help of X-ray crystallography and NMR, we can find the 3D organization of protein or protein complex. The interactions between proteins can be determined by protein microarrays.

Microarrays [2]:

Another important application which uses computational biology is managing complex experimental data. The best domain is microarrays where this kind of statistics is collected. When the experimental data is complex, there are two types of issues which need to be solved:

Firstly, the data which is considered has to be reformed so that it can be used by machine learning algorithms. This process is known as pre-processing. Secondly, data analysis is required to know the problem.

The most typical applications are identification of expression pattern, classification, and general network induction.

Systems biology:

It is a field where machine learning and biology work and organized with each other. It is complex to model the life processes that takes residence inside a cell. Thus, computational methods are very helpful to model biological networks, metabolic pathways, and single transduction network.

Evolution:

This domain particularly phylogenetic tree reconstruction uses the features of machine learning techniques. Phylogenetic trees are schematic representation of evolution of organism. Initially, they were constructed using the different features like morphological and metabolic features. Later, due to the huge amount of available genome sequence, the construction of phylogenetic tree algorithm used the concept based on genomes comparison. With the help of optimization techniques, a comparison was done by means of multiple sequence alignment.

Text mining:

Computational techniques have a side effect due to increasing amount of data, so text mining is a technique which is used for extraction of knowledge. Text mining is an interesting topic in computational biology which is applied in prediction of cellular location, analysis of protein interaction, and many other areas.

2 Unsupervised Techniques in Bioinformatics [3]

Unsupervised techniques are generally used to extract the organizational structure of the data, for example, the pattern which exists in gene expression of cancer. The unsupervised algorithm tries to discover the dominant recurrent features which are present in the database. They can be mainly susceptible to confounding factors.

The different methods of unsupervised learning are:

- Hierarchical clustering
- Partition clustering
- Model-based clustering.

2.1 Hierarchical Clustering [4, 5]

It is a type of clustering algorithms which divide the objects into a tree structure which consists of nodes, and each individual node denotes a cluster. Each node has any number or no child nodes.

Partition Clustering

In this type of clustering, the algorithm partitions the data objects into subsets that are not overlapping with each other in such a way that each subset has exactly one object. Each subset is called one cluster.

Model-Based Clustering

Models are used by these clustering algorithms where attempts are made to fit the data and the models. In case of model-based clustering algorithm, we consider that the data are coming from different clusters according to probability distribution.

Various unsupervised techniques in bioinformatics are as given below [2]:

- **Protein secondary structure prediction:** In this, we predict the three-dimensional protein structure that is the folding and its secondary as well as tertiary structure from the primary structure. Neural networks and support vector machines are used to perform this task.
- **Gene recognition:** It is a method of recognizing the areas of genomic DNA that encode genes. Protein-coded genes and RNA genes are included here. It may also contain predicting the functional elements. Hidden Markov model is used to solve this.
- **Multiple sequence alignment:** It is the process of aligning a sequence set which generally consists of the biological sequence protein, DNA or RNA. The techniques used to solve this are clustering and hidden Markov model.
- **Splice site recognition:** While technology is used to sequence the genome, a vast amount of sequence data has been created. For genome sequence, one of the main tasks is to identify all the genes. In case of eukaryotic genes, the coding region is depended on the identification of the exon-intron structures. Introns are non-coding regions of protein whose biological sequence is not yet known. The borders between exons and introns are known as splice sites. There are two splice sites, one in the upstream part of the intron and the other in the downstream part. The part of the intron which is upstream is called donor splice site, and the part of the intron which is downstream is called acceptor splice site. These two splice sites along with their relevant beginning and end of intron are called canonical splice sites.

There are numerous techniques to detect splice size. Markov model is used in the initial stage, and SVM is used in the next stage.

- **Microarray data—Normalization:** Many biologists are using the technology of microarray to monitor the genome-wide expression level of genes in any organism. Microarray is basically a glass slide on which the DNA molecules are fixed in a particular order at a specific location which is known as spots or features. Each DNA

molecule has thousands of spots which contain few millions of duplicate DNA molecules that individually correspond to a gene. The spots are printed on a glass slide with the help of a robot or synthesized by a process called photolithography. In order to detect differentially expressed genes, we require a reasonable measuring level that should not change in two different conditions. Generally, we notice that an average ratio of such genes differs by 1 which may be due to various conditions. In case of microarray experiments, as for large-scale experiments, many sources are there that affect the measurement of gene expression level.

Normalization is a technique of allowing such variations to compare data obtained from two different samples. The first step of normalization procedure is to select a gene set which consists of genes whose expression ratio is expected to be 1. Later, calculate normalization factor. Then, we apply this normalization factor to other genes in the microarray experiment. Since the normalization process changes the data, it is carried out in the background corrected value of each spot.

- **Microarray data—Gene selection:** In order to study the gene expression, sample classification needs to be done for which selection of gene is an important task. Researchers try to recognize the smallest set of genes that are used for prediction performance, e.g., to diagnose clinical abnormalities. Univariate ranking is used for significance of gene and arbitrary thresholds for selecting the number of genes. The gene selection ranking criteria are used to solve various classification algorithms. Some common techniques for gene selection are using classification algorithm like random forest or feature selection algorithm using a correlation-based feature selector combined with ML.
- **Microarray data—Prediction of therapy outcome:** Predictions of therapy outcomes can be done in two ways. The first method is to take the predictions directly from the patients, therapists, and clinical observers, whereas the second method is to take predictive measures from the same sources. There are various diseases worldwide, and one most common is depression disorder. Many options for treatment are available, e.g., neurostimulation and pharmacological, but there is no universal effective treatment. In this case, it is very important to identify the factors predicting a priori or early in treatment response or potential resistance. Most of the evidence which comes from research is proved statistically, but clinically they may not be always useful. “Clinical Predictor” should be cheap, dependable, reproducible, non-invasive, and easily accessible so that they can be used daily for clinical purposes. There are also many methodological issues combined with predictor or biomarker research.
- **Microarray data—Dependencies between genes:** The dependency of a gene with another gene plays an important role as it helps us to understand the different biological mechanisms. The interaction measures that already exist are based on association measures such as Pearson correlations. It can capture monotonic or linear dependency relationships. For nonlinear combinatorial dependency relationships, hidden Markov model is used. It is generally solved with the help of independent component analysis and clustering [6].

- **Protein structure and function classification:** A huge amount of sequence of data is produced by genome projects that have to be interpreted in terms of molecular structure and biological functions.

These projects have initiated many additional things like structural genomics [7–9]. The main intention of it is to determine the maximum number of protein structures in an efficient way and to explore the solved structures of biological functions to hypothetical proteins. It generally uses support vector machines and recurrent networks.

- **Alternative splice site recognition:** The complexity of a gene expression is increased with the help of a technique known as alternate splicing. This technique plays an important role in cellular differentiation and the development of the organism. The regulations used for alternate splicing are a complicated one where various interactive components are guided by functional coupling between transcriptions and splicing. Cancer is one of those diseases where we use the technique of alternate splicing. The techniques used are support vector machines and recurrent nets.
- **Prediction of nucleosome positions:** All eukaryotic organisms of DNA consist of nucleosomes which is the functional unit of chromatin. The accessibility of the DNA sequences needs to be changed in order to alter the accessibility of the DNA sequence. To study the genomic control mechanisms, the factors which have an impact on the nucleosome positioning are very important.

- **Single-nucleotide polymorphism (SNP):** Single-nucleotide polymorphism, as a rule abbreviated to SNP, is a substitution of a single nucleotide that happens at a specified function within the genome, and here each and every variation is present to some considerable measure inside a population.

For illustration, at a specified function of the human genome, for most contributors the C nucleotide could occur; however, in a minority of contributors, A can arise. This means that there is a SNP at a designated role, and the two viable nucleotide variations C and A are said to be alleles for this position.

Peptide and protein arrays: Peptide and protein microarrays are the rising instruments for proteomics and have excessive throughput ways that help to monitor binding events and events on a huge scale. Microarrays are used in epitope mapping and serodiagnostic applications.

On the basis of their application, protein microarrays are categorized as:

- (1) Analytical protein microarrays
- (2) Functional proteins, and
- (3) Reverse-phase protein microarrays.

Analytical protein microarray: The most representative model of analytical protein microarray is antibody array. Using direct protein labeling, proteins are found after antibody capture. For illustration, the sandwich assay layout employs two one-of-a-kind antibodies to search out the special protein, allowing differentiation between two similar blood samples.

Functional protein microarray: This type of array is created for the use of proteins that are purified individually. These types of protein array are used to

study the different biochemical properties of protein. This may include the study of binding activities of protein as well as enzyme–substrate relationships.

Reverse-phase protein microarrays (RPMA). It is a sort of microarray that identifies the different proteins by checking out various probes of lysate samples. At the beginning, some of these microarrays had been utilized in monitoring prostate cancer patients to recognize historical alterations.

- **Systems biology and modeling:** It is the theory of demonstrating the dynamics of biochemical links where molecules are the nodes and the connections between them are the edges. An explicit mathematical description of the connections and its interactive dynamics is used which helps in checking and predicting the performance of computer simulations.

2.2 Partitional Clustering with Respect to Genomics

In this type of clustering, the objects are partitioned by an iterative method into k clusters based on their dissimilarity as shown in Fig. 1. J is a predefined criterion function which assigns data into the k th number set. Due to this, result of the function clustering is possible. Partitioning methods are very useful for bioinformatics applications.

The different types of partitional clustering are as follows:

- i. **k -Means clustering:** [10, 11]

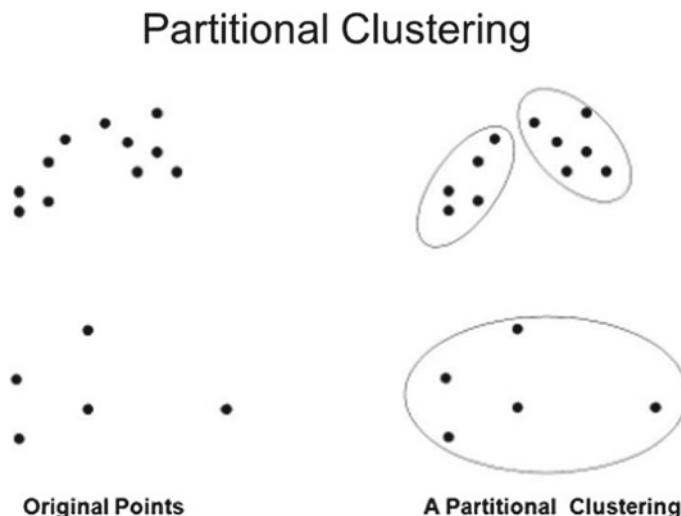


Fig. 1 Partitional clustering

It is one of the basic unsupervised learning procedures that are used to resolve the clustering problem. There are numerous ways to cluster the data, but k -Means algorithm is used maximum number of times which tries to increase the inter-group resemblance but at the same time keeps the groups away from each other. This algorithm functions on distance calculations where “Euclidean distance” is calculated. Euclidean distance calculates the distance between two given points using the following formula:

$$\text{Euclidean Distance} = \sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2}$$

This method is used to calculate the distance in two-dimensional space, but the same concept is also used for multi-dimensional space by expanding the number of terms. In case of k -Means clustering, “ k ” represents the number of clusters in which we want our data to divide into. The limitation for k -Means algorithm is that the data should be continuous. This algorithm will not work for categorical data.

If segmentation of population of India is to be done and the height is in cm and weight in kg, one can understand that the distance metric discussed above is highly susceptible to the units of variables. Therefore, it is recommended to make all the data in standard form before the clustering process.

k -Means is an iterative process of clustering, which keeps iterating until it reaches the best solution or clusters in our problem space.

The aim of this algorithm is to minimize an objective function known as squared error function given by

$$J(V) = \sum_{i=1}^c \sum_{j=1}^{c_i} (\|x_i - v_j\|)^2$$

where

“ $\|x_i - v_j\|$ ” is the Euclidean distance between x_i and v_j .

“ c_i ” is the number of data points in i th cluster.

“ c ” is the number of cluster centers.

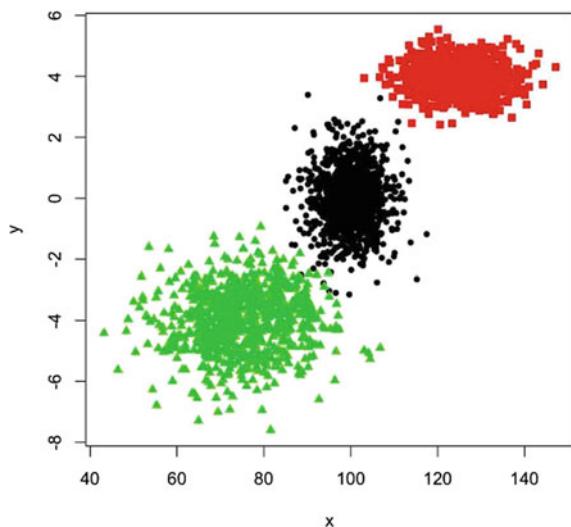
k-Means clustering algorithm:

Let the data points set $X = \{x_1, x_2, x_3, \dots, x_n\}$ and the set of centers be $V = \{v_1, v_2, \dots, v_c\}$ as shown in Fig. 2

- (1) In a random manner, select some cluster centers “ c ”.
- (2) Evaluate the distance between the cluster centers and data points.
- (3) Whichever data point has the minimum distance from all the cluster centers is assigned the cluster center.
- (4) The equation below helps to evaluate the new cluster center:

$$v_i = (1/c_i) \sum_{j=1}^{c_i} x_i$$

Fig. 2 Result of k -Means
for “ N ” = 60 and “ c ” = 3



where “ c_i ” represents the number of data points in i th cluster.

- (5) The intermediate distance between the data points and the new cluster centers needs to be again calculated.
- (6) We repeat this process of recalculation till no more data points are assigned; otherwise, from Step 3, we repeat the process.

Advantages

- (1) Fast, robust, and easier to understand.
- (2) In case of distinct dataset, we get the optimum result since the datasets are separated from each other.

Disadvantages

- (1) The number of cluster centers should be mentioned in advance for this algorithm.
- (2) k -Means algorithm is unable to resolve two clusters whose data is highly overlapping.
- (3) This algorithm gives different results for data which is represented in different ways, e.g., data in Cartesian and polar coordinates will give different results.
- (4) Haphazardly selecting the center of the cluster will not give us accurate results.

ii. Farthest first traversal k -center (FFT) algorithm:

In 1985, Hochbaum and Shmoys proposed this algorithm. The technique used in this algorithm is similar to k -Means clustering. Initially, it selects the centroids and assigns objects in clusters with maximum distance so that the initial seed values are far from each other. The working of the algorithm is as given below:

At the first stage, the initial data points are defined, and on the basis of that “ k ”, clusters are formed. At any time t where $t = 2, 3, 4, \dots, k$ select the data points from

the exiting center of the clusters and make it the t th cluster center. Thus, we see that the objects of dataset belong to cluster from Eq. 1 below:

$$\text{Min}\{\text{max_dist}(p_i, p_1), \text{max_dist}(p_i, p_2) \dots\} \quad (1)$$

iii. **k -Medoids or PAM:** [12]

This is a clustering algorithm significant to the k -Means clustering algorithm. This is a traditional partitioning technique of clustering which is used to cluster the dataset of n objects into k clusters. Medoid is an object of a cluster which is similar to almost all objects in the cluster. It is generally located at the central point of the cluster.

iv. **Clustering Large Applications (CLARA):**

In 1990, Kaufman and Rousseeuw designed a clustering algorithm which was used for clustering large application (CLARA) [13, 14]. It is basically an extension of k -Medoids method to decrease the computing time and storage problem because it consists of a huge number of objects, i.e., more than several thousands. This technique uses the approach of sampling and so instead of calculating the medoids for the full dataset, small sample datasets are considered, and the PAM algorithm is used to find the optimal set of medoids for this sample data [15, 16].

Resultant medoids value is calculated by the average difference among every object in the whole dataset and the medoid of its cluster, which is defined as the cost function. This process of sampling and clustering is repeated many times to reduce the effect of sampling. Finally, we get a collection of medoids whose cost is minimum.

The CLARA algorithm is as follows:

Step 1: From the actual dataset, randomly create a number of subsets with fixed size of the sample.

Step 2: On every subset, compute the PAM algorithm and find the corresponding k representative objects which are known as medoids. All observation of the full dataset should be assigned to the set nearest to the medoid.

Step 3: The difference between their observations and the medoid closest to it is calculated, and then, we find the mean of all these differences. This technique helps us to find the quality of clustering.

Step 4: The dataset which has the minimum mean value is preserved, and again an analysis is carried out on the absolute partition.

v. **Fuzzy k -Means:**

It is precisely the identical algorithm as k -Means, which is a fundamental and easy clustering technique. The basic difference is that in case of k -Means clustering, all clusters have specific points, but in case of fuzzy, some type of overlap is there to some extent, so a point can belong to two or more clusters. The important points for fuzzy k -Means clustering are as follows:

- k -Means clusters pursue hard clusters where a particular point belongs to one and only one cluster, whereas fuzzy k -Means clustering pursues soft clusters due to overlapping.
- In case of fuzzy clustering, a single point can belong to two or more clusters with some affinity to each of the points.
- The rate of affinity is directly proportional to the distance of that point from the centroid of that cluster.
- Like k -Means clustering, it also works on objects whose distance is distinct and can be defined in n-dimensional vector space.

vi. **k -Modes:**

It is a discrete version of k -Means with related runtime, welfares, and disadvantages. Clustering forms a crucial part analysis of data. One of the well-known and widely used unsupervised *learning algorithms* is *k -Means algorithm*. It can be used in various ways, for example, we use *k -Means to find the optimal nodal centers for Radial Basis function*. In this similar type of algorithm called “ *K -Modes algorithm*,” we use *modes* instead of *means* to form clusters of categorical data.

The k -Mode algorithm is as given below:

Step 1: Consider some arbitrary “ k ” number of data points which are called modes.

Step 2: Calculate the amount of differences between the selected data point and the remaining points.

Step 3: Data points are associated with those modes with minimum score. In this case, k clusters are formed.

Step 4: Using the technique “*Moving mode frequency based method*,” we change the modes and update the modes for all the clusters.

Step 5: Step 2 is repeated till there is no reassignment, and the cost function becomes minimum.

vii. **COOLCAT:**

Barbara et al. [17] proposed it which deals with k -Mode sensitivity. The sensitivity depends on the way the objects are selected. It is an incremental algorithm which decreases the entropy of the clusters. For example, if a set of clusters is given, then COOLCAT will put the next point in such a way that the entire entropy decreases. It is accomplished to cluster each and every new point without clustering the entire set. It is suitable for processing data streams.

Case Study 1 [18]:

An illustrative example shows the working of k -Means clustering. Figure 3 shows the eleven points in the example which has to be clustered.

Given a set

$S = \{A(11, 12), B(12, 11), C(13, 16), D(14, 8), E(8, 3), F(7, 2), G(7, 6), H(1, 8), I(2, 8), J(0, 7), K(0, 15)\}$ of points to be grouped into three clusters, and every point is getting weighted to $w = 0$; let $j = 0$;

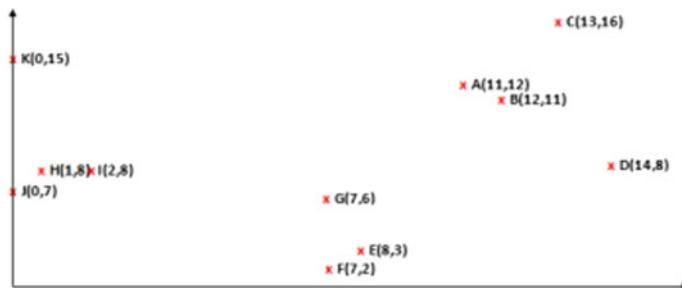


Fig. 3 Set of points to be grouped in three clusters

Step 1: The number of clusters $K = 3$. Weights of array should be as given below:

Point	Closest of	Its weight
A	B,C	2
B	A,D	2
C	\emptyset	0
D	\emptyset	0
E	F,G	2
F	E	1
G	\emptyset	0
H	I,J,K	3
I	H	1
J	\emptyset	0
K	\emptyset	0

Point	Closest of	Its weight
H	I,J,K	3
A	B,C	2
B	A,D	2
E	F,G	2
F	E	1
I	H	1
C	\emptyset	0
D	\emptyset	0
G	\emptyset	0
J	\emptyset	0
K	\emptyset	0

Step 2:
weights-
array in
descending
order

→

Step 2: Weights of array are arranged in descending order as shown above.

Step 3: C, D, G, J, and K are considered as outliers, since its weights = 0. The other points are considered as candidate points.

Step 4: The longest distance is between H and B. We can take B as a reference point.

Step 5: Distance-array should be as given below:

Point	Its distance to B
A	1.41
B	0
E	8.94
F	10.30
H	11.40
I	10.44

Step 6:
Distance-
array in
Ascending
order

→

Point	Its distance to B	Index
B	0	0
A	1.41	1
E	8.94	2
F	10.3	3
I	10.44	4
H	11.4	5

Step 6: Distance-array is arranged in ascending order as shown above.

Step 7: Difference-array should be as below:

differences	Index
$1.41 - 0 = 1.41$	0
$8.94 - 1.41 = 7.53$	1
$10.3 - 8.94 = 1.36$	2
$10.44 - 10.3 = 0.14$	3
$11.4 - 10.44 = 0.96$	4

→
Step 8:
Differences
-array in
Descending
order
→

differences	Index
7.53	1
1.41	0
1.36	2
0.96	4
0.14	3

Step 8: Difference-array is arranged in descending order.

Step 9: Subdividing the first $K - 1$ values.

differences	Index
7.53	1
1.41	0

→

differences	Index
1.41	0
7.53	1

Step 10: The first primary cluster should contain:

Point	Its distance to B	Index
B	0	0

The second primary cluster should contain:

Point	Its distance to B	Index
A	1.41	1

Step 11: The third primary cluster should contain the remaining points of distance-array:

Point	Its distance to B	Index
E	8.94	3
F	10.3	4
I	10.44	5
H	11.4	6

Step 12: Center of first primary cluster = Center of B(12, 11) = (12, 11); center of second primary cluster = Center of A(11, 12) = (11, 12); center of third primary cluster = Center of E(8, 3) F(7, 2) I(2, 8) and H(1, 8) = (4.5, 5.25).

Step 13: By the assistance of detected centers, we integrate k -Means algorithm to cluster all points of S . The partition will be: Cluster1 = {A, C}, Cluster2 = {B, D}, Cluster3 = {E, F, G, H, I, J, K}. Consequently, it is a bad result.

Step 14: Computing the result quality, and storing it in result-array.

Step 15: Based on weight-array in Step 2, the candidate points will be:

Point	Closest to	Its weight
H	I, J, K	3
A	B, C	2
B	A, D	2
E	F, G	2

F, G, I, C, D, J, and K are considered as outliers, since its weights $> j/j$ have been incremented to 1. Since the number of candidate points is greater than $K/K = 3$, we repeat the process from Step 4.

Step 5: Distance-array should be:

Point	Its distance to B	Point	Its distance to B	Index
A	1.41	B	0	0
B	0	A	1.41	1
E	8.94	E	8.94	2
H	11.40	H	11.4	3

→ Step 6:
Ascending
order →

Step 6: Arranging in ascending order.

Step 7: Difference-array should be:

differences	Index	differences	Index
1.41 - 0 = 1.41	0	7.53	1
8.94 - 1.41 = 7.53	1	2.46	2
11.4 - 8.94 = 2.46	2	1.41	0

→ Step 8:
Descending
order →

Step 8: Arranging in descending order.

Step 9: Subdividing the first $K - 1$ values.

Differences	Index
7.53	1
2.46	2

Step 10: The first primary cluster should contain:

Point	Its distance to B	Index
B	0	0
A	1.41	1

The second primary cluster should contain:

Point	Its distance to B	Index
E	8.94	2

Step 11: The third primary cluster should contain the remaining points of distance-array:

Point	Its distance to B	Index
H	11.4	3

Step 12: Center of first primary cluster = Center of A(11, 12) and B(12, 11) = (11.5, 11.5); center of second primary cluster = Center of E(8, 3) = (8, 3); center of third primary cluster = Center of H(1, 8) = (1, 8).

Step 13: By integrating *k*-Means algorithm using these detected centers to cluster all points of S, the partition will be:

$$\text{Cluster1} = \{A, B, C, D\}, \text{Cluster2} = \{E, F, G\}, \\ \text{Cluster3} = \{H, I, J, K\}.$$

It is a good result.

Step 14: Computing the result quality, and storing it in result-array.

Step 15: Condition not verified, so we stop running.

Step 16: Choosing the best result from result-array.

Outline of the proposed approach

The purpose of Step 1, Step 2, and Step 3 is to detect the densest regions in the dataset. Every region should be represented by a candidate point; at least we can know if this region is more or less dense according to the weight of its candidate point; the greater the weight, the densest the region is. Conversely, every outlier point will have a weight equals to zero. But it is not enough, we need to detect the longest distance between two candidate points, and we use one of them as a reference point in order to reveal the density and the remoteness among these candidate points, and that is what steps from Step 4 to Step 8 are all about. Steps from Step 9 to Step 12 are to discover primary clusters and their centers. Step 13 aims to integrate *k*-Means,

based on these detected centers. By using this strategy, we ensure that the initial choice of centers is reasonable, since these centers are remote from each other and they are in the densest regions. Consequently, the drawback of k -Means has been fixed up by adopting this new strategy.

Case Study 2

Experiments and Results

We present different genomic dataset as well as the evaluation measures used in our experiments; then, we display the obtained results and theirs comparison; finally, a discussion is conducted to evaluate the correctness and the efficiency of the proposed approach.

Dataset used in experiments

To measure the enactment of the proposed method, an experimental study was conducted using 20 different publicly available gene expression datasets, having the properties shown in Table 1. The analysis of the gene expression medium can be done in two ways, firstly by considering gene as data objects and secondly by treating the samples as data objects. In our empirical studies, we adopted the second strategy, i.e., the clustering of samples. The importance of this clustering helps to diagnose the disease condition, and it reveals the outcome of few treatment on genes [19].

Evaluation procedures incorporated in experiments:

One of important tasks of clustering is how to calculate results, without secondary information. A methodology for estimation of grouping results is to use validity indexes.

Clustering validity approaches can use external evaluation measure like F -measure and internal evaluation measure like Davies–Bouldin index.

- (1) External evaluation measure: It is a comparison between the obtained result and the expected result.
- (2) Internal evaluation measure: Other approaches measure the quality of generated clusters from the concept of “homogeneity and separation.” They are also defined to measure to what degree the data objects are similar inside one cluster, while dissimilar between different clusters. One common technique is Davies–Bouldin index method.

Empirical results and comparison:

Since all these algorithms are stochastic, we performed multiple runs over all 20 benchmarks, and each value is the average of 50 runs. The effects of F -measure, obtained by each algorithm, are shown in Table 2.

The Davies–Bouldin index which we get from each algorithm is shown in Table 3. As the clusters are close to each other, the DB measure is lower, whereas if the clusters

Table 1 Genomic dataset used in evaluation

Dataset name	Tissue	Total samples	Class number	Sample per class	Total genes
Alizadeh-v2	Blood	62	3	42, 9, 11	2093
Alizadeh-v3	Blood	62	4	21, 21, 9, 11	2093
Armstrong-v1	Blood	72	2	24, 48	1081
Armstrong-v2	Blood	72	3	24, 20, 28	2194
Bredel	Brain	50	3	31, 14, 5	1739
Chen	Liver	179	2	104, 75	85
Chowdary	Breast, colon	104	2	62, 42	182
Dyrskjot	Bladder	40	3	9, 20, 11	1203
Garber	Lung	66	4	17, 40, 4, 5	4553
Golub-v1	Bone Marrow	72	2	47, 25	1877
Golub-v2	Bone marrow	72	3	38, 9, 25	1877
khan	Multi-tissue	83	4	29, 11, 18, 25	1069
Laiho	Colon	37	2	8, 29	2202
Nutt-v3	Brain	22	2	7, 15	1152
Pomeroy-v1	Brain	34	2	25, 9	857
Pomeroy-v2	Brain	42	5	10, 10, 10, 4, 8	1379
Shipp-v1	Blood	77	2	58, 19	798
Singh	Prostate	102	2	50, 52	339
West	Breast	49	2	25, 24	1198
Yeoh-v1	Bone marrow	248	2	43, 205	2526

are spread out, then the measure is higher. Consequently, better quality cluster is obtained whose Davies–Bouldin index is minimum.

These experimental results have clearly revealed the difference before and after the improvement of k -Means algorithm. This new algorithm could be used in big data (millions of objects), and it is more likely to provide very good results, on one hand, since the problem of the local optimum had been fixed up, and on the other hand, the improved k -Means is simple to implement, fast, and easily parallelized.

3 Hierarchical Clustering with Respect to Genomics [8]

In order to do a comparative study of two datasets, we use scatterplots to give a visual representation. However, it is required for us to identify the group of genes with similar expression profiles for a number of experiments. The most common technique for finding the relationship among genes is cluster analysis. These clusters

Table 2 Results of *F*-measure

Benchmark	Improved <i>K</i> -Means	<i>K</i> -Means	<i>K</i> -Means++
Alizadeh-v2	1	0.8201	0.8518
Alizadeh-v3	0.7599	0.6771	0.6432
Armstrong-v1	0.6667	0.7215	0.7329
Armstrong-v2	0.7105	0.7567	0.8264
Bredel	0.7996	0.7024	0.6590
Chen	0.6446	0.8096	0.6895
Chowdary	0.6697	0.6697	0.6764
Dyrskjot	0.7990	0.7367	0.5376
Garber	0.6126	0.5784	0.5753
Golub-v1	0.7240	0.8460	0.8357
Golub-v2	0.8859	0.7789	0.8069
Khan	0.6926	0.6230	0.6031
Laiho	0.6771	0.7511	0.7289
Nutt-v3	1	0.7854	0.7045
Pomeroy-v1	0.6644	0.6750	0.6572
Pomeroy-v2	0.7358	0.6273	0.5859
Shipp-v1	0.7341	0.6891	0.7134
Singh	0.6286	0.6286	0.6286
West	0.6571	0.7437	0.6607
Yeoh-v1	0.9836	0.8991	0.8143

often suggest biochemical pathways. Mostly like mathematical tool, cluster study derives expressive results when they are mutual with biochemical insight.

The most commonly used mathematical technique is hierarchical clustering which tries to group small clusters and then the clusters into higher groups. We use dendrogram to view the resultant hierarchical tree structure. Most of the studies involve a series of experiments to identify the genes that are consistently coregulated due to certain circumstances such as disease state, increasing time, and increasing dose of drug. Each set of gene-expression levels is compared to all the other set of expression levels in a pairwise fashion, and similarity scores are produced in the form of statistical correlation coefficients. To make the correlations ordered, a node is created between highest-scoring pair of rows that is the geometrical closet. Then the matrix is modified where the joined elements are represented as a single node, and all the distances between the new node and other gene sequence of the matrix are computed. We do not recalculate the entire correlation but only change the rows which contain the new node changed values.

Typically, the dendrogram contains a link of the node, where the height of the relation is directly proportional to the strength of the correlation. This method of creating proportional links continues till all the genes in the experiment form a single

Table 3 Results of Davies–Bouldin index

Bench mark	Improved <i>K</i> -Means	<i>K</i> -Means	<i>K</i> -Means++
Alizadeh-v2	1.6366	2.2306	1.7122
Alizadeh-v3	1.6032	2.3249	1.7575
Armstrong-v1	1.9288	1.9636	1.9683
Armstrong-v2	2.0268	1.8721	2.1469
Bredel	2.1958	2.1241	1.8838
Chen	1.2488	2.4253	1.4717
Chowdary	0.8818	0.8819	0.8889
Dyrskjot	1.8901	1.7330	1.4242
Garber	1.7822	2.6147	1.7252
Golub-v1	1.8092	1.8891	1.8786
Golub-v2	1.7489	1.9410	1.9531
Khan	1.7493	1.9387	1.4035
Laiho	1.7147	1.8561	1.6953
Nutt-v3	1.7910	1.5726	1.6410
Pomeroy-v1	1.4546	1.6875	1.8476
Pomeroy-v2	1.4930	1.7259	1.6233
Shipp-v1	0.3683	1.4970	1.2275
Singh	0.8425	0.8425	0.8425
West	0.8815	2.0582	1.7026
Yeoh-v1	3.0918	2.5013	2.3834

hierarchical cluster where all the links are of appropriate length. In case two nodes are related to the same association, we can solve the problem by some predetermined set of rules.

In case of hierarchical clustering, the genes whose expression patterns are similar are put into a group which is connected with branches. This is known as clustering tree or dendrogram. The main task of hierarchical clustering algorithms is to partition all the objects into a tree structure as shown in Fig. 4.

The methods included in hierarchical clustering are:

Agglomerative: Originally, many minor clusters are formed which are combined based on their resemblance. At the end, a single cluster will contain all the objects.

- Divisive: Originally, all objects from one cluster are divided into smaller clusters.
- Steps in agglomerative clustering:

- Build the similarity matrix.
- From the similarity matrix, find the largest values.
- Combine the clusters having the largest value.
- Recalculate the matrix and iterate the process till all the clusters are combined together.

Hierarchical Clustering

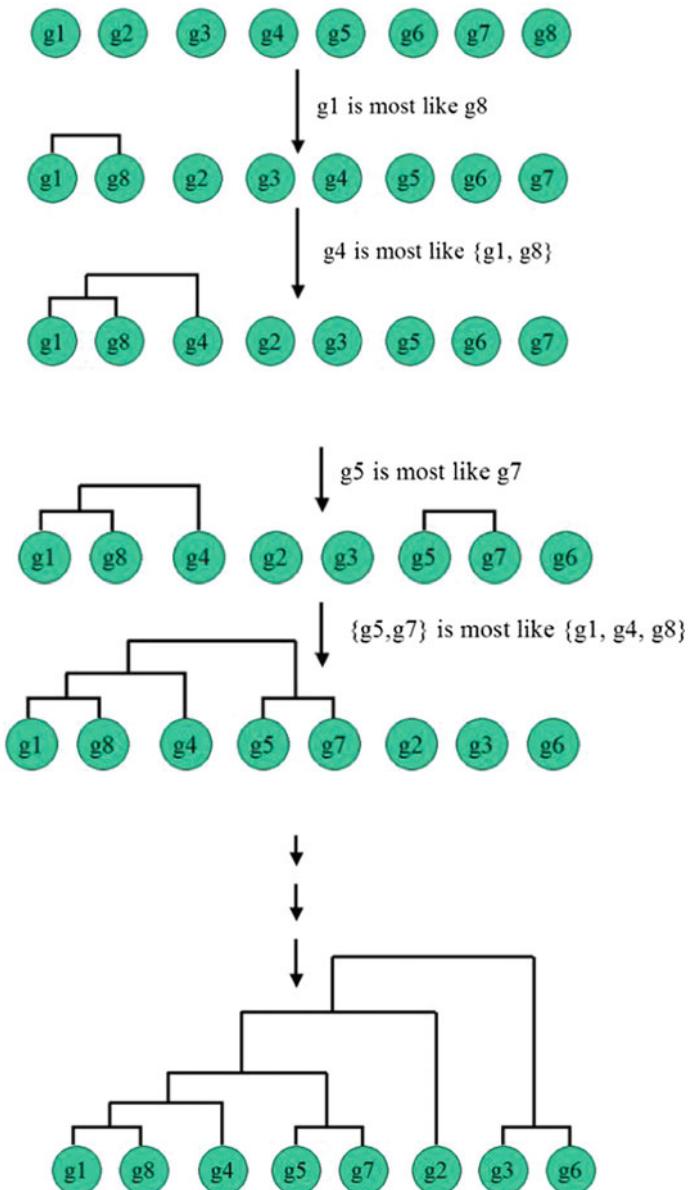


Fig. 4 Hierarchical clustering

With the help of linkage algorithms, we calculate the distance between clusters:

Single linkage: In case of single linkage clustering, the *distance* between two clusters is the smallest distance between any two members from two different clusters.

Complete linkage: In case complete linkage clustering, the maximum distance from any two members of two different clusters is the distance between two clusters.

Average linkage: In average linkage clustering, we measure the average distance from any members of two different clusters and the distance between two clusters $D(r, s)$ is calculated as:

$$D(r, s) = T_{rs} / (N_r * N_s)$$

where T_{rs} is the amount of all pairwise distances among cluster r and cluster s . N_r is the size of cluster r ; and N_s is the size of cluster s .

3.1 Advantages of Hierarchical Clustering

Types of experiments are generally used to recognize the overall similarities between gene expression patterns in the context of different treatment procedures—the aim is to stratify patients based on their molecular-level responses to the treatments. The hierarchical methods are suitable for such clustering, which is calculated on the pairwise statistical estimation of complete scatterplots instead of individual gene sequences.

3.2 Case Study 1: Charting Evolution Through Phylogenetic Trees

Hierarchical clustering helps us to relate different species. Decades before, DNA sequencing was a dependable technique, and scientists used to struggle to get relevant answers to simple questions for example. Are giant pandas closer to bears or raccoons? Nowadays, we use the technique hierarchical clustering and DNA sequencing to discovery phylogenetic tree of animal evolution. The steps are as follows:

1. Create the DNA sequencing.
2. Evaluate the “edit distance” which is a way to do a comparative study between all the sequences.
3. On the basis of the edit distances, calculate the DNA similarities.
4. Generate the phylogenetic tree.

As an outcome of this test, the scientists were able to place the giant pandas earlier to bears as shown in Fig. 5.

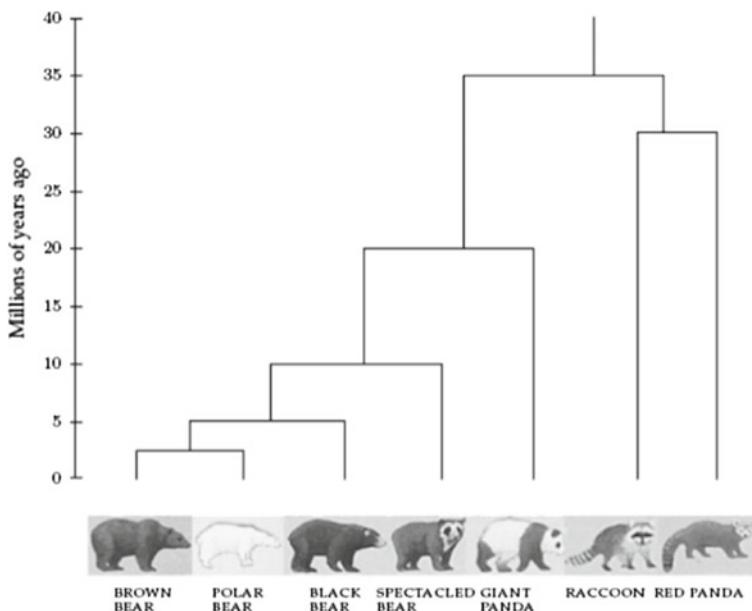


Fig. 5 Comparison of the giant pandas closer to bears

3.3 Case Study 2: Use of Hierarchical Clustering and Cluster Validation Indices for Analysis of Genetic Association [20]

It is generally considered that genes which are co-expressed propose co-regulation in the basic monitoring network. Defining the collections of co-expressed genes is a vital task which is created on few basic criteria of similarity. This task is done with the help of gathering procedures where the genes are collected into clusters on the basis of few experiments. A technique had been proposed to discover sets of co-expressed genes which are by validating clusters using indices as a measure of resemblance for individual group of genes and an arrangement of variants of hierarchical clustering to generate the candidate groups. We estimate its capacity to recover important sets on fake correlated and real genomics data where the performance is measured on its capability to recognize co-regulated sets against a full search. Further, we analyzed the value of the best placed groups using an online bioinformatics tool that gives network evidence for the nominated genes. The two indices used here to obtain a rank list of gene group are the Silhouette and Dunn indices which avoid extensive search but give very good and accurate results.

4 Conclusion

Thus, we see that with the help of machine learning, we can handle large amount of data. Since genomic data is relatively larger in size, machine learning approaches can simplify the things and make it easy to analyze. There are a lot of scope for genomics in machine learning like gene sequencing, gene editing, pharmacogenomics, etc. In case of gene editing, the genes are examined and then we discover the meticulous matches in genes and later change the gene sequence according to the way it needs to be targeted. Pharmacogenomics is extra field which offers more benefit for starting the modified medicine that is the drug which is assumed to the patient for a specific illness that should familiarize to the genetic makeup of the individual patient. Newly born inherited showing tools can make use of ML methods in classifying the metabolism defects. Thus, we can see that machine learning algorithms play a vital role in genomics.

References

1. http://www.iasri.res.in/sscnars/Genetics/14supervised%20classification%20_2_.pdf
2. <http://www.ijcstjournal.org/volume-5/issue-6/IJCST-V5I6P21.pdf>
3. Chiang D, Brown P, Eisen M (2001) Visualizing associations between genome sequences and gene expression data using genome-mean expression profile. Bioinformatic 17
4. Johnson SC, Hierarchical clustering schemes. Psychometrika 32
5. Dalton L, Ballarin V, Brun M (2009) Clustering algorithms: on learning, validation, performance, and applications to genomics. Curr Genomics 10(6):430–445
6. Rousseeuw P (1987) Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J Comput Appl Math 20(1):53–65
7. Sekhar SM, Siddesh GM, Manvi SS, Srinivasa KG (2019) Optimized focused web crawler with natural language processing based relevance measure in bioinformatics web sources. Cybern Inf Technol 19(2):146–158
8. Sekhar M, Sivagnanam R, Matt SG, Manvi SS, Gopalayengar SK (2019) Identification of essential proteins in yeast using mean weighted average and recursive feature elimination. Recent Pat Comput Sci 12(1):5–10
9. Patil SB, Sekhar SM, Siddesh GM, Manvi SS (2017) A method for predicting essential proteins using gene expression data. In: 2017 international conference on smart technologies for smart nation (SmartTechCon). IEEE, pp 1278–1281
10. Kenidra B, Benmohammed M (2016) An improved K-means algorithm for gene expression data clustering. In: Conference paper, Aug 2016, Research Gate
11. Chernoff H, A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. Ann Math Stat 195:493–509
12. Rondon E, Abundez I, Arizmendi A (2011) Internal versus external cluster validation indexes. Int J Comput Commun 5(1):27–34
13. Fred A, Jain A (2005) Combining multiple clusterings using evidence accumulation. IEEE Trans Pattern Anal Mach Intell 27(6):835–850
14. Topchy A, Jain AK, Punch W (2003) Combining multiple weak clusterings. In: Third IEEE international conference on data mining, pp 331–338
15. Lu Z, Peng Y, Xiao J (2008) From comparing clusterings to combining clusterings. In: Proceedings of the twenty-third AAAI conference on artificial intelligence. Institute of Computer Science and Technology, Peking University, Beijing

16. Strehl JGA (2002) Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *J Mach Learn Res* 3:583–617
17. Barbara D, Li Y, Couto J (2002) COOLCAT: an entropy-based algorithm for categorical clustering. In: Proceedings of the International Conference on Information and Knowledge Management, pp 582–589. <https://doi.org/10.1145/584792.584>
18. <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0184370>
19. Azuaje F (2002) A cluster validity framework for genome expression data. *Bioinformatics* 18(2):319–320
20. <http://www.informit.com/articles/article.aspx?p=357695&seqNum=4>

Supervised Techniques in Proteomics



Vasireddy Prabha Kiranmai, G. M. Siddesh and S. R. Manisekhar

1 Introduction

Bioinformatics is a hybrid science that integrates biological data with techniques of computer science, mathematics, and statistics to facilitate research in multiple areas like biomedicine, DNA and protein sequences, protein–nucleic acid complexes, etc. Bioinformatics is moving at a fast pace with respect to generation of data and information of biological processes, making its way in data storage, organization, and processing in data banks. The goal of bioinformatics is to develop efficient algorithms for sequence similarity calculations [1]. Dynamic programming methodologies, where full sequences are broken down into smaller sequence segments are used to find optimal pair of sequences. Needleman–Wunsch algorithm [2] is one such algorithm which detects gaps in sequence alignments. Microarray data and gene sequence data are solved using machine learning techniques of clustering, classification, and matrices theories.

Proteomics is a part of bioinformatics which extensively deals with the study of protein structure in the living organisms. The term proteomics came into existence in 1997 and is analogous to genomics, the study of genome. Proteome is the set of proteins produced by a system, and proteomics validates the identification of these proteins, with respect to time and stress changes in the organisms and uncovers the proteins structure, activity, and composition [3]. Proteome is a variant quantity differing from cell to cell with time and activity. Proteomics deals with expression of proteins, the rate of production and decomposition, science behind their modification, protein movement in sub-cellular cavity, protein interaction with each other, and protein involvement in the metabolism of the organism.

Protein is a macromolecular consisting of long chains of amino acids, which are a result of ribosome mechanism when it translates RNA from DNA in the cell nucleus,

V. P. Kiranmai (✉) · G. M. Siddesh · S. R. Manisekhar

Department of Information Science & Engineering, Ramaiah Institute of Technology, Bangalore, India

i.e., transformation from DNA to RNA and then finally proteins. Proteins can be classified based on the structural level as primary proteins which is an amino acid sequence consisting of twenty unit alphabet. And then, the secondary protein caused by the folding of amino acid sequences into alpha helices and beta sheets, tertiary proteins, which is a three-dimensional conformation of amino acids, and finally quaternary protein due to interactions between protein subunits to form a large unit of protein. The primary protein determines the secondary structure formation and the shape of the tertiary 3D conformation, which in turn determines the final structure of the protein conglomerate [4].

The study of protein, proteomics, can be further classified based on which aspect of protein is being studied. Structural proteomics deals with the study of the structure of the protein and atomic resolution of three-dimensional protein structures to understand protein sequencing. Expression proteomics [5] is the study of analysis and differential expression of proteins. It facilitates the measure of relative and absolute levels of proteins in a cell under simulation of time, drugs, and disease. Techniques used to label samples at the protein level and heavy oxygen usage at peptide level fall under expression proteomics. Interaction proteomics [6] deals with the interaction between proteins and activation of protein complexes. Biological insights and processes can be studied when interactions between the proteins are studied rather than individual proteins.

Proteomics is applied in a wide range of areas which include pharmaceutical drug discovery [7], proteomics in chemistry, protein complex analysis, excretion and metabolism, disease diagnosis and disease monitoring, etc. Drug screening is used for target identification and validation. Proteomics is being used in the drug industry since the past 20 years and it revolves around identifying proteins expressed between the control and disease samples. It is also used in protein expression profiling which plays a vital role in Pharmaceutical Research in which protein profiles are compared between the disease and control samples in the laboratory. These are called disease biomarkers. Cell surfaces and proteins are concentrated in the protein profiling technique. Proteomics in chemistry is targeted for compound screening which is used in the discovery of new drugs using target-based screening and phenotypic screening, which is based on phenotypes like cytokine contents from the cells. This technique can be adopted to produce high-quality drugs. Monitoring of drug proteins in tissues or blood can be used for Pharmaceutical Research and developments.

Machine learning techniques are being used to analyze data from biology fields like data generation by analytical methods of metabolism and transcriptomics for identification and classification of genes. These techniques are now even used in proteomics to analyze the data collected about proteins and mass spectrometry. Quantitative protein information can be obtained from biological sample by various machine learning techniques. Some of the most commonly used techniques are gel electrophoresis, affinity separation, mass spectrometry-based technologies, protein arrays, etc. Some of these techniques can only analyze a limited number of proteins and specific type of proteins due to time constraints. Mass spectrometry (MS) techniques give hyper-efficient list of proteins than the other techniques. This makes it more suitable for finding proteins which have not been considered before

and are not already listed in the protein database. Combination of other proteomics along with mass spectrometry can be used to find to form a pipeline of analysis of various proteins. Machine learning techniques like mass spectrometry can be used for exploring proteins using database searching, preprocessing, and protein quantitation.

Machine learning techniques are used for identifying biomarkers, which is a measurable indicator of biological condition. Size of the protein dataset is used in the selection of machine learning techniques that could be used to classify various proteins. Various algorithm-based techniques with bottom-up approaches integrated with machine learning methods are used in the analysis of proteins and protein identification. Mass spectrography is a widely used technique for protein identification using ionization and dissociation of them. Greater experimental requirements and more complex instruments along with modern machine learning techniques can be used to analyze a variety of proteins.

2 Scope of Data and Machine Learning in Proteomics

Massive growth in the amount of biological data requires efficient information management and storage along with extraction of useful information from the available data. Development of various methodologies and tools for converting the data into biological knowledge involves adoption of machine learning techniques to give rise to new models to uncover this information and provide predictions of the system. In proteomics field, computational methodologies, combinatorial techniques, and machine learning techniques are used in protein structure and function prediction and in management of complex experimental data. Firstly, protein-related data needs to be preprocessed so that it is made suitable for applying machine learning algorithms. Next, data needs to be analyzed based on what has to be predicted using pattern identification [8].

Genome of any organism consists of patterns of thousands of proteins which is the root cause of function of life. Flawed proteins result in diseases such as diabetes, cancer, and dementia. Hence, the study of proteins plays a major role in drug recommendations. To understand diseases and processes of life and to provide appropriate treatment, it is important to analyze proteins. Mass spectrometry is used to determine the type and quantity of proteins in biological systems. Proteomic data can be used to train neural network which is used to recognize proteins in lesser time and greater accuracy [9].

Machine learning in proteomics is specifically applied in the classification of sample of various proteins and identification and investigation of potential biomarkers. Biomarkers are quantifiable characteristics of biological processes, which do not relate to the sense of well-being of the patient but rather help in improving the drug development process and contribute to biological research by application of various techniques [10]. This could be performed using laboratory-based and computational techniques. Machine learning analysis in proteomics requires large datasets, and it

is essential to consider the types of protein samples available and their sustainability to apply machine learning techniques [11].

Increase in the amount of data that is available provides sources of information where different mining techniques can be applied for extraction of knowledge from the available data. This can be done using the text mining techniques. Data and text mining has become very useful in computational biology and it has a wide range of applications and can be put into use in cellular location prediction, functional annotations, and protein interaction analysis [8]. Computational techniques of data mining can be used to solve back translation of proteins which is a complex combinatorial problem.

Machine learning is used in proteomics to optimize the performance by using data from datasets and examples. Optimization can help in improving the degree of accuracy obtained in prediction models in case of modeling problem. Statistical methods are used to build computational models to make computers learn from the datasets and process this data to represent the model. Efficiency of the Learning algorithm used should have high accuracy and considerate space and time complexity. Transformation of data to knowledge is iterative and interactive process.

In the iterative phase, data integration and merging of information from various sources are done followed by data cleaning, where elimination of incorrect data takes place. Selection of relevant variables present in the data, data mining, inconsistencies, and outliers will be resolved and the appropriate technique for the data analysis, either supervised or unsupervised classification will be selected. Among these techniques of machine learning, various models are studied to understand which one suits the data that we possess. After the model is chosen, it is evaluated from both the computational and biological perspectives. This step is repeated till an efficient model is acquired to compute the available protein data.

Various approaches and tools based on a wide range of algorithms are developed. There are several classification and biomarker analysis techniques which are very specific to the proteomics domain [12]. In addition to these approaches, the conventional machine learning techniques like KNN algorithm, logistic regression, support vector machine (SVM), decision tree algorithms, and neural network can be applied. These techniques fall under supervised learning techniques. Unsupervised learning techniques like clustering and probabilistic graphical methods like Bayesian networks are also used in proteomics. Data preprocessing techniques like wavelet and genetic algorithms are used for feature extraction and feature selection which is a vital step for protein classification.

3 Machine Learning in Proteomics

Supervised machine learning is used for classification of proteins. A data model is developed from training data which has labels for each of the sample present in the data. This model is used to identify the classes of various data samples present in the dataset which does not have a label. Classes could range from disease group or

phenotypes or treatments. The attributes present under each class could be identified proteins [2]. A general classification problem has a set of attributes divided into classes. The attributes have some features associated to them and a set of classification rules which help in distinction of these various attributes under various classes. Supervised classification paradigms are algorithms which induce classification rules from data in the dataset which could be used in classifying data which is out of the dataset [8].

Various supervised learning algorithms like Bayesian classifier, rule-based learners, decision trees, random forests, and support vector machines can be applied in protein classification techniques, and biomarkers identification initial step would be to collect the data which can be analyzed and studied further. Datasets can be pre-processed and required features can be extracted using feature selection techniques and later they can be provided as input to the known machine learning algorithms according to our specifications.

In this section, we first discuss the structure of the proteomic datasets, some of the proteomic datasets which are available, and later on how feature extraction can be performed on the available dataset, and later we analyze the various supervised techniques present.

3.1 Proteomic Dataset

Proteomics dataset is a data matrix which can be represented X_{ij} having a response variable in the form of protein consisting of i rows which are the complete observations and j columns which are proteins. “Mass unit” is the term used to represent proteins or mass over charge (m/z) units. When the dataset is subject to classification techniques, an additional dummy variable vector is considered, Y , which is used to identify group membership of the observations, which is called the indicator matrix [13].

3.2 Sample Datasets Available for Proteomics

A wide range of techniques can be applied to the available proteomic datasets. Each of the dataset represents a number of diseases and mass spectrometric platforms. Some of the examples of the datasets which can be used are.

3.2.1 Lung Cancer and Colorectal Cancer (CRC) Dataset

This dataset consists of 50 cancer cases with 50 and 45, respectively. These datasets have undergone variable preselection, and there are 39 variables in lung cancer dataset and 109 variables in colorectal cancer dataset. These datasets have been acquired

by MALDI-TOF/MS technology and these datasets have been preprocessed. These datasets can be used in classification algorithms.

3.2.2 Ovarian Cancer (OC) Dataset

This dataset has been acquired by MALDI-TOF/MS technique, and it consists of 24,262 features. This dataset has undergone appropriate preprocessing steps. There are various methods of classification like KNN, aggregated classifiers, SVM, and quadratic discriminant analysis that can be applied on this dataset. Variable importance like univariate t-statistic is used for variable selection, and this creates 15 and 25 variable datasets. These reduced datasets can be analyzed using classification models.

3.2.3 Gaucher Disease Dataset

This dataset is obtained by SELDI-TOF/MS technique from serum of 20 Gaucher disease. Potential outlier is removed from this dataset. This dataset consists of 590 variables and has been preprocessed. PCA-LDA technique is adopted by the authors for data classification, and the predictive ability is demonstrated by double cross-validation approach.

The variables present in the dataset are prefiltered or selected to be provided as the input to the classification methods. There are various preselection and preprocessing techniques that can be performed on this dataset. Preselection is performed using some of the known techniques like linear models which uses a moderated t or f-statistic to select the appropriate features. In addition to this, data preprocessing also plays a major role in the elimination of unnecessary data from the datasets and this preprocessed data can be used in preselection of the significant variables [13].

3.3 Data Preprocessing Algorithms

Datasets are preprocessed by removing the relevant data from the dataset. Data preprocessing plays a vital role in knowledge extraction from the datasets [12]. Proteomic dataset contains large amount of data which needs to be preprocessed and feature selection to pick the important and significant attributes are to be applied. Some of the important data preprocessing algorithms that are applied to the proteomics dataset are.

3.3.1 Wavelet Algorithm

This algorithm can be used in removing the noise associated with the dataset and also feature selection for preprocessing of the protein classification. Wavelet transformation is used in the data analysis of the biological data which highlights the application of useful wavelets in biology. Wavelet analysis is used in preprocessing the data to reduce the data dimensions. Discrete wavelet transformation is applied after data binning and is used in further compression of the dimensions of the data. This technique has been applied to 121 cancer samples, having dimensions of over 370,000. Using this method, it has been reduced to 6757 and further reduced to 3382 features. Various wavelet techniques have been presented for processing MS data transformations. Stationary discrete wavelet transform is another technique used for de-noising of the dataset and reduction of dimension from 20,000 to 1800 on a mice tumor dataset. Further, support vector machine technique has been applied to the reduced features of this dataset, and this has produced an accuracy of over 99%. Wavelet-based method can be used to preprocess mass spectrometry dataset which consists of heterogeneous noise. The performance of detections has been shown to improve by applying the local wavelet thresholding of data. Another wavelet technique, dual-tree complex wavelet transformation, uses symmetric Hilbert-pair of wavelets which is also used to de-noise the data and has been proven to perform better than discrete wavelet transform and stationary wavelet transform.

3.3.2 Genetic Algorithm

This algorithm is an optimization procedure which is an iterative approach of populating of candidates using natural selection to solve an objective function. A dataset which is dimensionally large is chosen, and feature selection is applied by genetic algorithm. This technique is applied to a feature selection algorithm for pyrolysis mass spectrometry. This algorithm is used to find optimal subset of regression variables for various models like multiple linear regression and partial least squares. By applying this method, variables in the dataset have been reduced from 150 to 20 variables. A range of algorithms have been developed by combining genetic algorithm with other algorithms like cluster analysis methods. This technique is used to detect early-stage cancer. Genetic algorithm is integrated with Mahalanobis distance, and it was used to classify disease spectral samples and normal spectra samples. Datasets containing 3077, 12886 and 74263 features have been preprocessed with a classification accuracy of 92.6%. Another method called best genetic algorithm is the best in prediction. This method is tested on DSI dataset containing 162 ovarian cancer samples and 91 control samples, having accuracy with 25th and 75th percentile with 97 and 99%, respectively. Quick classifier, support vector machine, and genetic algorithm are integrated into software to generate models, which are used to analyze the data collected and categorized. Through various series of experiments, the models constructed optimized genetic algorithm with 100% recognition, 99% cross-validation achievements, and 100% positive predictive value. Another technique is

developed by the integration of genetic algorithm and Bayesian network for identification and classification of *Bacillus* species. The integrated technique is used to reduce the variables from 150 to 22 on the subset of the data. This technique was effective in discovering the biomarkers for spores. Mathematical model of genetic algorithm and support vector machine is to select the peptide peak. Classification techniques are used to determine optimal separating hyperplane for classification. This technique is also used in serum peptide classification with high accuracy using MALDI-TOF system. Another multivariate analysis based on genetic algorithm is with principal component analysis which is linear discriminant analysis. This is used in the lipidomic approach for blood plasma classification.

Using these techniques, data can be preprocessed and the dimensionality of the data can be reduced significantly which helps in faster and more accurate processing of datasets.

3.4 Dimension and Feature Subset Selection

Dimension reduction is the method in which there is a reduction in the number of variables that are taken into consideration. This is done by only picking important and principle variables [14]. This procedure helps in reducing the number of features in the dataset without having to lose the required information and affect the performance of the model. This is very powerful in case we are dealing with a large dataset.

There are various techniques for dimension reduction, low variance filter, high correlation filter, random forest, backward feature elimination, backward feature elimination, principle component analysis, independent component analysis, forward feature selection, and many other techniques [15]. With respect to the proteomics, principal components analysis (PCA), partial least squares (PLS), and linear discriminant analysis (LDA) are significantly used.

3.4.1 Principal Component Analysis (PCA)

It is a widely used dimension reduction technique in huge and complex datasets. The goal of this technique is to summarize the data into lesser dimension without losing any important information. This is accomplished using matrix techniques of mathematics. A data matrix is represented by X_{ij} , which is a product of two matrices, score matrix T_{ik} and loadings matrix P_{ij} . This technique is often represented as k-PCA which means the number of components or variables which are extracted from the data, such that a data point is represented in a k-dimensional space. Mathematically, it is represented by,

$$X = TP^k + E$$

where E stands for the residual error.

Principal component analysis aims on constructing a linear combination of the original variables which are linearly independent which are orthogonal to each other. Euclidean distance is taken into consideration for finding the distance between various observations. The original observations are projected onto a new latent variable, and then the distance between the observations and the new latent variable dimensional space is calculated.

3.4.2 Partial Least Square (PLS)

This is a colonial projection technique, which provides supervised dimension reduction capacity. This method is often used for datasets which have class membership variable represented by y and predictor variable x . This is different compared to other dimensional reduction techniques like principal component analysis as this algorithm calculates latent variable from x based on y . It aims at increasing the covariance between X and y , whereas PCA maximizes the variance of only X . This algorithm takes into covariance of these two variables rather than just variance of one variable. The latent variables are product of interactive decomposition of X and y causing the original variables to project to a lower dimension resulting in dimensional reduction. Mathematically, it is represented by,

$$\begin{aligned} X &= TP^k + Ex \\ \text{and } y &= TC^k + Ey \end{aligned}$$

where T represents the scores of the latent variables that the data is projected to, P and C are loads, and Ex and Ey are residual matrices obtained from original data X and y .

The above two methods provide class separation at a qualitative level, they are not classification techniques but purely dimension reduction technique. They are generally used with some classification methods.

3.4.3 Linear Discriminant Analysis (LDA)

This technique is a linear combination of all the new components obtained from the principal component analysis and partial least square on dimensional reduction process. The disadvantage of using LDA is that it is not able to deal with $n \ll p$ type of datasets. In such cases, use PCA or PLS and then later apply classification technique rather than using a formal approach like LDA. The technique that LDA uses projects the known observations into a new coordinate system and then passes these values to a classifier. Later, a model is developed to predict the classes of the unknown observations, using the prior probabilities found from the learning set. Linear combination is used to calculate and maximize the ratios between within class

variance and between class variance. The assumption that LDA considers is that all the matrices are equal variance or covariance matrices and are normally distributed.

3.4.4 Feature Subset Selection

Generally, dimension reduction is done in the feature subset selection process. This is an advantageous step to proceed with supervised classification stage as it will reduce the cost of data acquisition, improve the overall efficiency of the classification model and faster classification model, and increase classifier accuracy. This problem of feature subset selection can be described as a search problem, where each state in the search space specifies a subset of possible features. There is a large amount of computational data, and finding the possible feature subset is unfeasible. There are four basic steps to this, identify the search space starting point, search organization, subset evaluation function, and halting criteria. Initially, the direction of search is chosen, the features get added one by one starting from an empty set. The search organization strategizes the search in the space of size of the feature subset. The evaluation function is used to measure the effectiveness of the subset of features that have been selected. The halting criterion is used to stop the search for subset space identification. An effective subset of the observations would help in proper optimization of time and space, helping in the overall efficiency of the classification problem.

3.5 Protein Classification

Protein classification [16] is accountable for biological sequences, and classification of proteomics can be done using machine learning techniques. Sequences are classified in the form of protein vector which is used in representing of proteomics. Feature selection where important feature is selected is highly challenging, along with high accuracy. Initially, protein space construction and analysis is performed which plays a wide role for protein classification using machine learning techniques.

3.5.1 Protein Space Construction

Protein sets which are related having similar structures or functions belong to the same protein group. A family of proteins which consist of higher classifier called G-protein receptor is the basis for which they are divided. Family of proteins is classified as family and domain which are used in classification of proteins. These terms have been come up with biologists for computational purposes. There are various aspects like description of protein family, protein domain architectures, and species distribution which play a vital role in classification of proteins. Classification is the step after the feature extraction step in which important and necessary features

are extracted. Various approaches can be used for classification like support vector machines, which is a regression classification based on the primary structure of the proteins and the family they belong to. Using this technique, a sequence of protein is created which is a distributed representation of proteins. The dataset is divided into training data and test data, in which the system is trained using the training dataset and then the test data is fed into the system and verified. The algorithms are chosen based on the datasets and the accuracy that is required. Generally, a large amount of training data is required for training the machine to learn and based on the information present in the dataset. The protein sequence machine learns and outputs which sequence the input data belong to within a very limited period of time with high accuracy based on the classification method chosen. Some of the algorithms used are n-gram which uses the information from the protein for lapping of the window from three to six residues. Using this technique, more accurate results are obtained for different window sizes.

3.5.2 Protein Space Analysis

Protein space analysis is used in the training space for analyzing the physical and chemical properties that are being used in n-grams classification. Some techniques use other properties like volume and mass for classification of the data. The protein space characteristics are studies from Lipschitz constant which is a mathematical concept which deals with uniform continuity of function and how fast function changes with time. In our context, it deals with how the physical and chemical compositions of proteins are changing quickly with time and other biological changes.

4 Supervised Algorithms in Proteomics

4.1 Decision Trees

Decision trees structure the extracted data or information and discriminate the data in a tree-like structure. Decision trees improve the understandability of the classification and identification of attributes and classes [11]. This algorithm depends on the statements which are conditional and unconditional and uses some available tools to make a decision. These tools can analyze graph-like models and understand them, where the problem has a set of possible outcomes and the chance that the outcome occurs. Based on this information, these tools evaluate the event that is most likely to happen [16]. There are a variety of decision trees, and simple decision trees are the easiest of all, where the classification and the various relevant branches can be understood with ease. Methods like C4.5 are also used in proteomics. In this technique, we start with an empty tree and the data is split in iterative fashion, then branches are created and till all the data is classified into either of the trees branches based on

criteria, this method is iterated. Each of the data points is a tree leaf in decision tree technique.

Decision tree technique fits well in proteomics problems. Computational complexity of decision trees is linear in the number of input variables in the worst case scenario. It performs considerably well when the input sample is large compared to the number of samples, where most of the variables are irrelevant. Its complexity is faster. This technique can be used in analyzing biomarkers. This algorithm can be combined with any ensemble approach which gives rise to more advanced algorithms, which have hybrid computational and functional properties yielding better analysis results [17].

In proteomics, decision trees have been widely used in the past for classification purposes with a high percentage of accuracy and precision [12]. In 2002, Baoling Adam has come up with a proteomic classification using the decision tree algorithm. This is based on the nine-protein mass pattern. This experiment is done with the blood samples of two types—prostate cancer antigen (PCA) and healthy man cohort—to validate a system which showed 96% of classification accurately. A year later, Markey used classification and regression Tree model is to classify 41 clinical specimens with an accuracy of 90%. Size of the data plays a crucial role in interpretation of results. Later in 2012, advanced decision tree was used to develop sequence identification algorithm, and this has significantly improved the accuracy and precision. In 2013, cleavage prediction with decision trees has been invented and was widely used in mass spectrometry-based peptides data. This algorithm helps in reduction of search space and time complexity. Later in 2018, predictive models based on decision tree algorithm were used for the analysis of *Staphylococcus aureus* strain. The average accuracy of all the above example is more than 83%.

4.2 Support Vector Machine

Support vector machines are the type of machine learning which are based on prediction using the linear separability between classes [11]. It is a computationally effective supervised model and is based on statistical learning. It establishes a plane to classify various patterns in the classes. This algorithm depends on the linear separability between classes. A hyperplane used in classification is based on a transformation method and kernel function where an input feature vector space is transformed into high-dimensional space [18]. This feature makes SVMs solve complex classification between sets.

In a practical scenario, the data points in the dataset are not linearly separable in input space. Then, the kernel function comes into work. The data points should not be overfitting, and so kernel function projects the input space to a higher-dimensional space. Kernels function is based on the Mercers' condition. This function maps input space to a higher dimension reducing the possibility of overfitting. There are a variety of kernel functions, polynomial function, Gaussian radial basis kernel function, etc. SVM technique is based on support vectors which are elements of the training set

based on which the decision boundaries of the classifier are set. Proteomics dataset contains the training examples which are very difficult to classify. SVM uses these support vectors to classify and predict the samples present in the dataset.

Support vector machine algorithm is used in proteomics due to its ability to handle high-dimensional data using a transformation function to convert to low-dimensional data. One of its applications is in analysis of ovarian cancer biomarker discovery and classification, which had details about the 37 patients who were suffering from papillary ovarian cancer. The quantization of the data is done using mzMine technique and SVMs along with some feature selection methods is used in classification. This way, an accuracy of 83% is achieved when nonlinear SVM is used along with LOO-CV technique and an accuracy of 97% is obtained when nonlinear SVM is used along with SVM-based feature selection techniques [8].

4.3 Random Forest

Random forest is based on the decision tree algorithm, wherein multiple trees are built over the training data. Every tree is associated with a sampled subset of attributes related to the problem [11]. Each individual tree predicts a class based on the training set, and a random forest is built using all of these decision trees, and the prediction classification of the random forest is based on the prediction of each individual tree. Random forests are a learning technique used for both classification and regression.

In a practical scenario, let us consider that there are N training sets and we use a bootstrap method, which is the mean sample with some replacements in the training dataset. The taring set would be two-third to the number of the original dataset. Then, we would construct a classification and regression tree (CART) [1] for every bootstrap dataset and this would result in the formation of decision tree for each of the bootstrap set. If there are M features in each of the input vector, then m more relevant features are chosen for the classification purpose. All the results of each bootstrap dataset classification are aggregated resulting in the formation of the final classification of the random forest.

In proteomics which involves a large number of features in the dataset, multiple decisions trees can be built and be classified using the random forest algorithm considering only the relevant and import features in each dataset. In 2003, the performance of classification of ovarian cancer dataset is calculated using classifiers, which include bagging and boosting, SV, and random forest algorithm. Using random forest along with feature classification of the features in the dataset, the accuracy of the net-ore classification has increased to 92%. In 2004, random forest algorithm is applied with 1000 trees on the data with a sample of 100 spectra, each spectrum containing 138 peaks, and the error rate have come to 32%, sensitivity to 76%, and specificity to 64%. Later, standardization and de-noising of the datasets are used to classify the data of whole-organism bacterial specimens. Using this technique along random forest has improved the accuracy of the entire classification.

In 2008, random forest [12] was implemented to classify the proteomic profiles obtained by mass spectra. The dataset has been collected over 76 breast cancer patients. Random forest classified these samples into target classes with a specificity of about 86% and sensitivity of 82%. Later in 2010, random forest was used in phosphorylation dataset. A nonlinear random forest classification along with discrete mapping approach technique is used in classification. In 2014, random forest was used in classification of *N*-glycopeptides using mass spectral features. In 2016, SVM and random forest were used to classify the geographical originals of 31 white rice samples, and accuracy of 93.66% and 93.83%, respectively. In 2017, mass spectrometry along with random forest and SVM with radial basis kernel, C5.0, average neural network, and kNN has yielded superior results with accuracy of 95%. Random forest model is more robust to outliers and negligees overfitting making it more suitable for using it against large datasets.

4.4 Logistic Regression

Logistic regression is a regression technique where the variables are categorical. Let us consider there are multiple parameters present in the model. All of these parameters should be estimated from the data to build a concrete model. Parameter estimation is performed using maximum likelihood estimation method [8]. Estimations are obtained using iterative manner. Newton–Raphson procedure is a standard technique in obtaining estimations. The modeling process is based on the likelihood ratio test and Wald test, and the search in the space of models is done using forward, backward, or stepwise approaches.

In proteomics, logistic regression is used in cases where most classification algorithms cannot give a precise prediction. In 2004, a model was proposed based on regression models, and better efficiency and robustness were achieved using a better coefficient vector of regression model. The coefficients of the regression are computed using SVD decomposition of the datasets. In 2009, a new method called Oseore is introduced which is developed by logistic regression based on the training dataset. The dataset is based on the 18 protein mixtures. This technique estimates the probability of the correct peptide in the mass spectrometer spectrum. In 2017, logistic regression model is used for mass spectra data in proteomics case-control studies. Using the logistic regression technique in proteomics guarantees great calculation speed and consistency in the results obtained along with an optimal solution.

4.5 K-Nearest Neighbor

The nearest neighbor algorithm is used to classify a given data point to a label based on the nearest data points to the test data point. This is the basis of the k-nearest

neighbor algorithm. In k-nearest neighbor algorithm, a label is assigned to the test-point based on the k-nearest samples of the dataset [8]. Put in other words, means that the classification of any data point of assigning a label to it is based on the labels or the characteristics of the k nearest neighbors of the test data point.

In a practical scenario, any problem can be solved using the k-nearest neighbor algorithm. But the efficiency or time complexity of the algorithm is directly proportional to the size of the dataset. The larger the dataset, the more time it is required for the classification of the dataset. A strategy that could be used in order to minimize the slowness is offered by this algorithm to classify each example with respect to the examples in the dataset which are already seen and save only those that are misclassified. This approach of solving the classification problem using kNN algorithm is called condensing.

This algorithm is used widely in proteomics for the classification of the proteomic dataset which is very large. This is the easiest algorithm to be applied to the dataset. In 2003, k-nearest neighbor algorithm is compared with the other traditional classification algorithms, like linear discriminant analysis, bagging and boosting, classification trees, SVMs, random forest, and quadratic discriminant analysis [12]. It is observed that k-nearest neighbor algorithm performed as good as the other algorithms which made it more reliable to be used in proteomic datasets. This algorithm was used in classification of a dataset which consists of MS spectra, and the data for this dataset was acquired from 47 patients with ovarian cancer and 44 normal patients. Later, KNN algorithm was improved using Mahalanobis distance to classify lung cancer dataset, and KNN algorithm gave better results compared to the other algorithms, like linear discrimination method, by reducing the misclassification rate. In 2004, KNN algorithm was integrated with other algorithms like genetic algorithm (GA), and this algorithm has better classification results with an accuracy of 96%.

Later in 2015, KNN algorithm is integrated with lasso regression to apply to a mass spectrometric image dataset in detecting the gastric cancer tissue, and this algorithm has reduced the classification error to 3%. Later came the idea of “customized training” to apply to the subset of the test data, in which a customized training technique is used to select a subset that is close to the subset that has to be classified. In 2016, KNN was implemented using the Euclidean distance technique formula and this technique proved to produce amazing results with the spectrometer generated data. Without the use of clinical records, classification of the spectrometer data produced an accuracy of 75% when KNN algorithm was used. In 2018, various algorithms resulting in classification are compared to compute the missing values of the metabolic datasets.

4.6 Classification Trees

This algorithm classifies a pattern by asking and answering a set of questions related to the sequencing of the data points, till a directed classification tree is obtained, this tree is classed the classification tree, where there is a root node is located at the

top and is connected or directionally linked to the other nodes [8]. This linkage is done from the root till the leaf nodes are reached in the tree. The classification of a pattern is started from the top of the tree and it goes on till the property of the pattern is achieved. The different paths traversed in the tree lead to different patterns that can be derived from the classification tree. In a classification tree, the links between the different nodes are exhaustive, which means that only one link would be there to traverse from one node to the other node in the tree. The further step would be to make a decision at a subsequent node, which would be the root of the subsequent subtree which is chosen. This technique is adopted till a leaf node is reached. The leaf node consists of a categorical label associated with it, and the pattern that is traversed in the tree is assigned to the label obtained on traversing.

The classification tree would progressively split the training labeled dataset into smaller and smaller datasets. When an ideal situation is considered, the samples of the subset would bear the same categorical label. If this is the scenario, then the subset is considered to be pure and the process of dividing the subset further would be terminated. But in a practical scenario, there are mixtures of labels in each subset of the classification tree. So, before splitting the subset further, we might have to accept some misclassification and some impurity in the decision instead of splitting it into much smaller trees. This is a recursive tree-growing process, from the root to the leaves of the trees to keep finding the perfect node of the tree, where there could be a possible and best split off the tree. Ratios like, Gini Ratio and Gain ratio is used to compute the best and appropriate variable at each level of the tree where there could be a split done.

In proteomics, this technique is widely used in classification of the datasets at appropriate variables by the formation of the classification tree from the entire datasets. In 2003, a new classification tree technique was adopted to discriminate proteins using mass spectrometry [12]. This is obtained by projecting the dataset using the wavelet transformation and then using the feature space construction by using feature selection technique. The recursive classification tree algorithm can be used to partition the feature space by dividing the sample into much smaller samples. In 2011, a classification tree model was used to discriminate the patients with pulmonary tuberculosis from patients who were non-tuberculosis with a sensitivity of 98% and specificity of 85%. Later in 2014, a classification decision tree was proposed which is used to classify acute leukemia spectra which was further divided into five subsample groups. A maximum tree was obtained with a root node, and then the proteomic-based classification was found to be consistent with the MIC-based classification technique. A comparatively good accuracy rate was obtained when the classification technique was adopted.

5 Applications of Supervised Learning Algorithms in Proteomics

Machine learning algorithms have played a vital role in proteomics with respect to classification of various proteins. Various machine learning algorithms like nearest neighbors were used in the past for the prediction of the secondary structure of the proteins. Then, another method, classification tree is used for the prediction of the protein secondary structure present in it. Later, two-stage methods involving support vector machine and a Bayesian classifier are used to predict the surface residues present in proteins which are widely visible in the case of protein–protein interactions [8]. Another problem related to prediction of the protein sub-cellular locations automatically using only the sequence has been analyzed using the fuzzy k-nearest neighbor algorithm.

Another application where machine learning in proteomics plays a vital role is in the extraction and analysis of the genetic information of the Human Genome Project [16], and this topic is of importance as it is blooming with respect to the number of researches held with respect to it. The analysis of the proteins and the structures of the various proteins is specifically used in the protein purification and mass spectrometry.

Here, we briefly discuss how various machine learning supervised techniques are used in the field of proteins in analysis of them and understanding the proteins better and in solving some problems in this field using the classification and statistical models.

5.1 *Proteomic Mass Spectra Classification Using Decision Tree Technique [17]*

Mass spectrometry is used for generating protein profiles of various body fluids like saliva, urine, or serum. These measurements help in the diagnosis of various diseases as the structure of the proteins would vary between the diseased and the normal patients. It also helps in monitoring the response of patients to various medicines or drugs. The data acquired from the patients are typically high in dimensions consisting of several thousand variables, with less number of samples. In these cases, classical statistical techniques like linear discriminants and neural networks can be used to reduce the dimensions of the data. An alternative to these techniques is to use the advanced machine learning techniques like kernel-based methods and tree-based ensembles to explore the datasets without any prior feature extraction and elimination.

5.1.1 Problem

The main objective of this section is to identify biomarkers of a particular disease from proteomics MS datasets, by discriminating between a certain class of the disease or by recording the responses to a particular treatment. Predictive models are constructed to explore the biomarker present in the datasets, helping in diagnosis of the diseases.

5.1.2 Dataset Collection

The datasets are collected from the past biological samples of various patients and are classified based on various factors and are later processed by mass spectrometer. Mass spectrometer is used to provide more accuracy to the data using the signal intensities. The number of variables or the dimension of the data acquired is very large based on the number of patients. Usage of the machine learning algorithm is decided based on the number of dimensions of the dataset.

This experiment is conducted using two datasets of SELDI-TOF-MS obtained from the serum samples of the patients. The motto of this experiment is to detect patients suffering from inflammatory disease. These samples were collected from the University Hospital of Liege from 2002.

5.1.3 Machine Learning Requirement Analysis

Various techniques can be used to extract information from the datasets, and supervised learning technique is also applied based on the samples described by the input variables and the output information. The goal of the entire process is to extract a synthetic model which can predict the output information from the input variables. The development of learning algorithm is used in the construction of the classification model. Various algorithms can be used for this process, like neural networks, discriminant analysis, or decision tree. Based on the specificities of the application, an algorithm picked is based on the number of samples, how informative the variables are, in order to determine the biomarkers.

Decision tree techniques used in these problems fit well in these characteristics. The complexity of these methods is linear with respect to the number of the input variables, and they cope when input space dimensionality is greater than the number of the samples, as many input variables are not relevant. It has a better time complexity compared to all other supervised algorithms. It is exploratory approach, and hence, subset of the important variable is identified easily, i.e., biomarkers. The basic decision tree can be integrated with another ensemble approaches like bagging and boosting, producing better algorithms which has hybrid computational and functional properties.

5.1.4 Method

The data which is noisy even after the mass spectrometer cleaning should be eliminated or filter out the noise before applying any machine learning technique on it. This is removed using simple m/z discretization algorithm. Using a small ratio m/z ratio discretization provides cleaner data points.

Model Construction

In case of single decision tree, CART algorithm is used with cost-complexity pruning. Several decision trees along with ensemble methods are used to reduce variance and bias, improving accuracy and reliability. Notable studies have been conducted to compare various tree-based ensemble methods, and no method has been found to outperform another. In this problem, four methods are applied in parallel and the results of the experiments described are:

Bagging: Tree is built based on CART algorithm from bootstrap samples and all the tree predictions are aggregated using majority technique.

Random forest: This technique is derived from bagging, and at every node, k attributes were chosen, where the split of the tree is determined.

Extra trees: A complete learning set is selected at each node by selecting the best among k randomly generated splits.

Boosting: A tree is built sequentially using CART, by increasing the weights of the learning set samples.

Model Cross-validation and Selection

In training the learning algorithm, each sample is removed from the learning set till reasonable accuracy is acquired. Then, from the remaining $n - 1$ samples, another sample is eliminated. In practical scenarios, selection of the best among several models is done based on the cost of misclassification and the error rate. The factors on which the decision should be taken are:

Sensitivity: Percentage of sample from the target class that are well classified.

Specificity: Percentage of sample from the other class that are well classified.

Error rate: Percentage of sample that are misclassified by the model.

Based on the combination of the above three factors, the model is selected.

Biomarker Selection

The identification of biomarker is a procedure consisting of two steps one after the other: Firstly, the attributes are ranked in decreasing order of their significance, and later, a subset of biomarker is chosen using cross-validation.

Attribute Importance Ranking

The importance of the attributes is calculated using the various measures in the classification problem, and importance measure technique is adopted for this. Shannon entropy is computed using the class frequencies for computing the subset of the samples. Split is important as it discriminates between different classes.

Biomarker Selection

The most optimistic subset of biomarkers is selected based on the error estimates. A model is built with all the relevant attributes, and using machine learning algorithm, the best one is selected using cross-validation. Only on the most important attributes, algorithm is used. The accuracy of the model is computed so as to determine the curve of the model. The attribute which has the maximum accuracy is chosen as the biomarkers.

Validation

Sensitivities, specification, and error rate are widely used in eliminating the various attributes in the model classifications. The learning set consists of two or four replicas of each patient and that repeated data point has to be removed. Data preprocessing by peak selection gives good results. Boosting is applied to increase the superiority of the model generated. C4.5 is the base learning on top of which ensemble algorithms are used. Preprocessing of data is done using discretization. Various methods are compared like kNN and SVM, and the results of SVM are better, but the best trees-based method is better than SVM, making it the best suitable classification technique.

This application has demonstrated a flexible and systematic method from extraction of knowledge from the proteomic dataset. It has highlighted the usage of supervised machine learning algorithms like decision tree induction and various decision tree ensemble methods along with pre- and post-processing stage. Various tools and methods are used to extract data from the dataset, and biomarkers are identified to be used in clinical decisions.

5.2 Distance Metric Learning and Support Vector Machine for Classification of Mass Spectrometry Proteomics Data [19]

Mass spectrometry establishes a connection between biomedical diagnosis and identification of protein. A data sample involves a sequence of the ratios of mass/charge. The data mining consists of four steps, preprocessing, feature extraction, feature selection, and classification. The mass spectrometer data consists of noises like

electric noise and chemical noise which result in reducing the efficiency of the classification. The motto of the preprocessing of the data is to purify the data.

Feature selection is used to identify the relevant feature present in the dataset, and later support vector machine recursive feature elimination is used to select a small subset of the input attributes to classify and prepare the learning model. Distance metric learning technique is also used in the classification of the proteomic mass spectrometry data points. Later, the results of both the algorithms were compared and it is found that both of these algorithms produce equivalent results. SVM-RFE is chosen over SVM as the former involves feature subset selection using recursive approaches, filtering out only the most important attributes.

5.2.1 Dataset

These are the two datasets that are used in experimentation:

High-resolution time-of-flight mass proteomics dataset is acquired from the 121 ovarian cancer cases and 95 controls. The dataset have been fetched from FDA-NCI clinical proteomics.

Breast cancer QC SELDI data has been fetched from 57 controls and 51 cases.

The dataset is processed by the peak detection. Later, large margin nearest neighbor (LMNN) classification is applied to the detected data. Support vector machine recursive feature elimination is a feature selection technique which uses the weights of the support vectors in classification of the proteomics data using SVM method. Later, the results are compared with LMNN technique which is based on Euclidean distance, Mahalanobis distance, and classification based on energy for detecting of the peak data. For each of the sample, 80% of the data is declared as the training dataset and the reaming 20% is the data used for testing. Each of the experiment is repeated for 10–100 times and the average testing results are declared.

5.2.2 Comparison of Results

For the ovarian cancer dataset, when the classifier LMNN using the energy way of classification is used, an accuracy of about 99.3% is achieved. While when LMNN algorithm using the Euclidean distance technique is used, accuracy of about 84.6% is used. Further, when LMNN technique using Mahalanobis distance is used, an accuracy of 99% is acquired. When the same set of algorithms are used against breast cancer dataset, the accuracy is calculated. LMNN with energy way of classification provides an accuracy of 81.8% and when LMNN technique using Euclidean distance is used, then an accuracy of 84.6% is achieved, later when LMNN using Mahalanobis distance is used, an accuracy of 81.7% is achieved. This has proven that LMNN using energy way of classification provides the best accuracy in model classification compared to the other LMNN techniques.

Comparing the results of these two classification approaches, we can conclude that applying SVM to SVM-REF feature datasets fetch us better results and applying

LMNN classifier based on energy classification and Mahalanobis distance yield better results than applying SVM to SVM-RFE feature sets. LMNN classifiers with energy classification and Mahalanobis metric have superior results in the classification of proteomics data. Later, classification results have been compared when LMNN is applied to feature sets chosen by SVMRFE, and it has been concluded that the lesser number of features yield better results in each experiment compared to the use of SVM.

This experiment compared two algorithms picked up from supervised distance metric learning, nearest neighbor classifier and support vector machines for the classification of mass spectrometry proteomic dataset. Comparing the results of the two algorithms that have been applied, it is concluded that applying distance metric learning algorithm to the proteomic dataset have yield superior results in comparison to support vector machine. Further, this experiment can be extended to the use of distance metric learning in feature selection for the datasets.

6 Conclusion

In the recent times, the main challenge in bioinformatics and proteomics is in transformation of the large amount of data present, into knowledge using the advanced technologies appropriate techniques. Statistical methods and machine learning algorithms serve this purpose and simplify the data backed with great accuracy.

This article has discussed the application of machine learning algorithms in the field of proteomics. The classification and identification of the samples of proteins and their structure were able to be studied. Biomarkers were able to be predicted using various supervised classification algorithms. Also, various machine learning techniques have been highlighted here and how and where these methodologies can be put into use along with various methods in purification of the dataset. Various feature selection and elimination techniques in case of large datasets have been discussed, where high-dimensional input data can be scaled down to a lower-dimensional data, where only the relevant and most important attributes are chosen. A variety of statistical and mathematical concepts that have been used in analysis of which attributes to be eliminated without any changes in the accuracy have been proposed in this paper. Protein space construction and analysis have been identified and studied, where subsets of the dataset have been chosen. Various supervised learning algorithms, like decision trees, support vector machine, random forest, and others, have been briefly described on how they have been used in the proteomic in the past highlighting the merits and demerits of each of the algorithm. Later, the important application of machine learning techniques have been briefed throwing some light on the application of theoretical algorithm in the actual field on clinical classification, involving identification of biomarkers and comparison of various supervised learning algorithms in a practical example.

References

1. Can T (2013) Introduction to bioinformatics. Part of the Methods in Molecular Biology book series (MIMB, vol 1107), pp 51–71. https://doi.org/10.1007/978-1-62703-748-8_4
2. Lesk AM (2019) Bioinformatics. <https://www.britannica.com/science/bioinformatics>
3. Yee A, Pardee K, Christendat D, Savchenko A, Edwards AM, Arrowsmith CH (2003) Structural proteomics: toward high-throughput structural biology as a tool in functional genomics. <https://doi.org/10.1021/ar010126g>
4. Introduction to Proteomics, Wikibooks. https://en.wikibooks.org/wiki/Proteomics/Introduction_to_Proteomics, 2017
5. Center for Proteomics and Bioinformatics, Expression Proteomics, Western Reserve University, Cleveland, Ohio. <http://proteomics.case.edu/proteomics/expression-proteomics.html>, 2010
6. Center for Proteomics and Bioinformatics, Interaction Proteomics, Western Reserve University, Cleveland, Ohio. <http://proteomics.case.edu/proteomics/interaction-proteomics.html>, 2010
7. Yokota H (2019) Applications of proteomics in pharmaceutical research and development. *Appl Proteomics Pharm Res Dev*
8. Larranaga P, Calvo B, Santana R, Bielza C, Galdiano J, Inza I, Lozano JA, Armananzas R, Santafe G, Perez A, Robles V (2005) Machine learning in bioinformatics. *Brief Bioinform*
9. Artificial intelligence boosts proteome research, Technical University of Munich (TUM). <https://www.sciencedaily.com/releases/2019/05/190529113044.htm>, 2019
10. Strimbu K, Tavel JA (2010) What is biomarker? *Curr Opin HIV AIDS* 5(6):463–466
11. Swan AL, Mobasher A, Allaway D, Liddell S, Bacardit J (2013) Application of machine learning to proteomics data: classification and biomarker identification in postgenomics biology. *OMICS: J Integr Biol*
12. Fan Z, Kong F, Zhou Y, Chen Y, Dai Y (2018) Intelligence algorithms for protein classification by mass spectrometry. *BioMed Res Int*
13. Sampson DL, Parker TJ, Upton Z, Hurst CP (2011) A comparison of methods for classifying clinical samples based on proteomics data: a case study for statistical and machine learning approaches. *PLoS ONE* 6(9):e24973. <https://doi.org/10.1371/journal.pone.0024973>
14. Dimensionality reduction, Wikipedia. https://en.wikipedia.org/wiki/Dimensionality_reduction, 2016
15. Sharma P (2018) The ultimate guide to 12 dimensionality reduction techniques (with Python codes). <https://www.analyticsvidhya.com/blog/2018/08/dimensionality-reduction-techniques-python/>
16. Naveenkumar KS, Mohammed Harun Babu R, Vinayakumar R, Soman KP (2018) Protein family classification using deep learning. Center for Computational Engineering and Networking (CEN)
17. Geurts P, Fillet M, de Seny D, Meuwis M-A, Malaise M, Merville M-P, Wehenkel L (2005) Proteomic mass spectra classification using decision tree based ensemble methods. Oxford Academic
18. He B, Zhang B (2013) Discovery of proteomics based on machine learning. Beihang University
19. Liu Q, Qiao M, Sung AH (2008) Distance metric learning and support vector machines for classification of mass spectrometry proteomics data. In: Seventh international conference on machine learning and applications

Visualizing Codon Usage Within and Across Genomes: Concepts and Tools



Bohdan Ostash and Maria Anisimova

1 Introduction

The sheer complexity and diversity of life forms on Earth hinge on a genetic code. The latter is set of rules that define how nucleotide sequence (information) is converted into proteins (workhorses of cellular metabolism) in course of ribosomal protein synthesis. In mRNA, four nucleotide bases can form 64 possible trinucleotide sequences or codons. Each codon corresponds to a specific amino acid (sense codons) or stop signal in the process of protein synthesis. There are 61 sense codons, while natural proteins are built of only 20 amino acids. Hence, the extant genetic code, theoretically, may encode three times as many different amino acids. Elucidation of the origin of the genetic code and driving forces behind the evolution of protein-coding sequences are of great fundamental and applied interest. Much has been achieved in these areas since the discovery of mRNA triplet structure over 50 years ago, leading to a refined set of algorithms and tools for the analysis of codon evolution [1]. Wide adoption of next-generation sequencing technologies since 2005 [2–4] has led to tremendous growth of nucleotide sequence databases, allowing application of these tools to virtually any biological problem. Larger datasets might indeed provide insights into function and evolution of coding sequences. Yet, they pose nontrivial challenges that require an educated choice of analytical tools and care in interpretation of results. As analysis of protein-coding sequences at DNA and amino acid

B. Ostash (✉)

Department of Genetics and Biotechnology, Ivan Franko National University of Lviv, Lviv 79005, Ukraine

e-mail: bohdan.ostash@lnu.edu.ua

M. Anisimova

School of Life Sciences and Facility Management, Institute of Applied Simulations, Zurich
University of Applied Sciences ZHAW, 8820 Wädenswil, Switzerland

Swiss Institute of Bioinformatics, Lausanne, Switzerland

levels historically dominated the field, for a non-expert it is not always obvious what advantages codon-oriented methods would bring. Therefore, modeling of sequence data of high volume and complexity requires, in our opinion, more user-friendly applications which would represent the data in a visually informative way. This should foster the growth of interest in codon-based studies and, most importantly, improve or even guide our understanding of a biological problem behind the numbers. In the latter, we believe our thoughts resonate with current state of the entire field of biomedical data analysis [5]. This chapter aims to introduce the reader to a specter of approaches available for the study of protein-coding sequences at a codon level. Below, we describe the basic concept of each approach, list relevant visualizations tools and outline possible directions for future development. Throughout the text, we emphasize the visualization part of a problem; the reader is referred to excellent and up-to-date literature where extensive treatment of conceptual and mathematical issues of certain approach is offered. We start with a description of the concept of the genetic code. Although being already a textbook paradigm, it remains an area of intense research and ingenious visualization efforts. Description of programs for k -mer analysis will lay the ground for several sections devoted to codon analysis tools—such as codon context discovery, codon indices, substitution models and methods of estimation of selective pressure. Future prospects and challenges for the codon-based studies and tools will conclude the chapter.

2 Genetic Code

An mRNA sequence is “read” by ribosome in consecutive non-overlapping nucleotide triplets, known as codons. Genetic code specifies a set of rules that assign each codon to one of 20 proteinogenic amino acids (Fig. 1). Three codons (UAG, UAA and UGA) are recognized by protein factors in course of protein synthesis and serve as translation termination signals (stop codons). The number of codons (64) exceeds the number of amino acids (20) used by cell to make proteins. Thus, the genetic code is degenerate—one amino acid can be encoded by more than one codon. The distribution of codons across amino acids is uneven: In the standard (or canonical—see next section) genetic code, three amino acids—Arg, Leu and Ser—can be encoded by as many as six codons, whereas only a single codon is assigned to Trp. Codons having the same “meaning” at the protein level translate to the same amino acid and are referred to as synonymous. The reasons for different degrees of synonymicity of different amino acids are obscure. However, it is worth mentioning that amino acids abundant in proteomes are encoded by a higher number of codons [6], and less abundant amino acids (His, Cys, Trp) are thought to be incorporated into the extant genetic code later during the evolution [6, 7]. Hence, evolutionary older amino acids could just have greater chance to be encoded by larger number of codons. Genetic code is also quasi-universal. That is, the same set of rules of translation operates in most of the organisms, with a notable number of exceptions; the latter will be briefly touched in the next section. Four codons having the same

		2 nd base in codon					
		U	C	A	G		
1st base in codon	U	Phe Phe Leu Leu	Ser Ser Ser Ser	Tyr Tyr STOP STOP	Cys Cys STOP Trp	U C A G	3rd base in codon
	C	Leu Leu Leu Leu	Pro Pro Pro Pro	His His Gln Gln	Arg Arg Arg Arg	U C A G	
	A	Ile Ile Ile Met	Thr Thr Thr Thr	Asn Asn Lys Lys	Ser Ser Arg Arg	U C A G	
	G	Val Val Val Val	Ala Ala Ala Ala	Asp Asp Glu Glu	Gly Gly Gly Gly	U C A G	

Fig. 1 Table of canonical genetic code provides information on the amino acid assigned to each codon. Initiator methionine codon is shown in green

first two letters and differing in the third one (synonymous position) are labeled as codon box. Split codon boxes are those encoding two amino acids (or amino acid(s) plus stop codon; see Fig. 1). Un-split codon boxes encode a single amino acid (e.g., Pro, Ala, Gly). The organization of genetic code table is far from being random. Particularly, amino acids similar in their physical and chemical properties tend to group together in the table. Second codon position is a key determinant of codon assignment. This largely confers error minimization property to the genetic code, as substitutions or mistranslation in synonymous position does not change codon meaning, while substitutions in the first position switch one amino acid to a related one. A number of other regularities in natural genetic code prompt additional theories about how these regularities might help maintain biological function, of which minimization of frameshift errors during translation is perhaps the most salient [8].

2.1 Canonical Genetic Code and Mechanism of Its Realization

The most widely used version of genetic code is referred to as universal, standard or canonical. Currently, over 20 variations of canonical genetic code are known [9, 10]. Reassignment of usual stop codons (UAA, UAG and UGA) to certain amino acids is typically encountered within noncanonical genetic codes. For example, in

the mammalian mitochondrial genome UGA is decoded as Trp, while the canonical Arg triplet (AGG) serves as a stop codon. In a few ciliate genomes, all stop codons are simultaneously used as sense codons, and termination of translation is context-dependent [11]. In some yeast species, canonical Leu codon CUG is reassigned to serine. However, in certain species, such as *Candida albicans* or *Ascoidea asiatica*, this codon can be read as both leucine and serine. This leads to a rather unique situation where Ser and Leu residues are incorporated randomly into protein. An mRNA carrying n CUG codons would lead to 2^n different proteins. This causes a stochastic heterogeneity of proteins within yeast cells (so-called statistical proteome [12]) and, consequently, phenotypic variation of the latter. The aforementioned examples attest to the evolvability of the genetic code [13].

The decoding of the mRNA is done by ribosomes and requires transfer RNAs (tRNAs). The latter are small (usually 76 nt), highly structured and post-transcriptionally modified entities serving as a bridge between nucleic acid and protein worlds. The aminoacyl stem (AAS) and anticodon stem loop (ASL) are two prominent parts of tRNA having distinct biological roles. AAS of each tRNA is specifically charged with the amino acid by aminoacyl-tRNA synthetases (ARS), leading to aa-tRNA. Ribosome provides an environment where formation of mini-helix between codon of mRNA and anticodon of cognate aa-tRNA will be promoted—a crucial checkpoint prior to the peptide bond formation [14]. In the vast majority of organisms, the number of tRNA species is fewer than the number of sense codons, e.g., 46 in *Escherichia coli*, 42 in *Saccharomyces cerevisiae* and 29 in *Mycoplasma capricolum*. Some of the tRNAs, referred to as isoacceptors, are able to recognize more than one synonymous codon. Sets of tRNAs having different anticodons and charged with the same amino acid are known as isoacceptor families. First two positions of a codon form Watson–Crick pairs with bases 36 and 35, respectively, of ASL (e.g., G·C, C·G, U·A, A·U). The third codon base, referred to as a wobble position, may form atypical hydrogen bonding with the 34th position of the anticodon (e.g., U·G, G·U, I·A/U/C), thanks to extensive modifications of heterocycle and carbohydrate portions of bases in 34th and 37th positions of ASL [15] (Fig. 2).

It has to be noted, however, that the importance of post-transcriptional tRNA modifications goes far beyond ensuring wobble interactions [16–18]. This notion is supported by an observation that even genomes carrying very large sets of tRNA genes (e.g., human cells express around 300 different cytoplasmic tRNAs) and possessing tRNA specific for each codon still maintain a sophisticated tRNA modification machinery [19, 20]. To summarize, three layers of code can be distinguished in the process of gene expression. The first one is the genetic code that determines the meaning of mRNA codons. The second one is embodied by 20 aminoacyl-tRNA synthetases that control correct charging of tRNAs. Post-transcriptional tRNA modifications constitute the third code for correct charging of tRNA, decoding and protein synthesis in general. For example, going back to the *C. albicans* case, the ambiguous reading of CUG codons in this species is enabled at two levels. First, the tRNA^{Ser}_{CAG} is charged with serine (97%) and leucine (3%) because both Ser-aRS and Leu-aRS

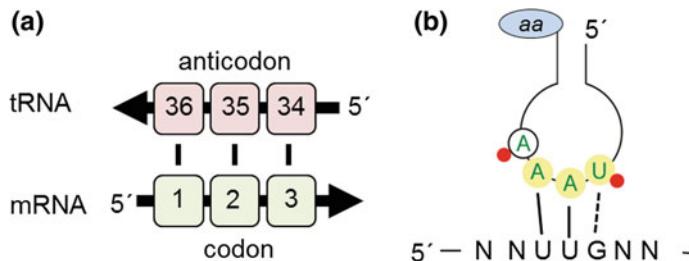


Fig. 2 Triplet structure of protein-coding genes: basic terms. **a** Three consecutive bases of codon are numbered as 1, 2 and 3. In tRNA, anticodon positions are 34, 35 and 36. Mini-helix formation between codon and cognate anticodon proceeds in antiparallel fashion; e.g., two-nucleotide strands run in opposite directions, 5' to 3'. **b** The correct codon–anticodon interaction is promoted and stabilized by post-transcriptional modifications (red ovals) of nucleoside residues within anticodon stem loop of tRNA. For example, modification of uridine within position 34 allows it to form non-Watson–Crick pair (dashed line) with guanine in wobble (third) position of codon UUG. Modifications of nucleosides outside the anticodon (such as position 37) are also important for the decoding process

recognize this tRNA as a correct one. This is because tRNA_{CAG}^{Ser} carries identity elements for both aRSs: ASL is specific for Leu-aRS, while D-arm and AAS are fit for Ser-aRS [21]. Second, guanosine residue downstream of anticodon of tRNA_{CAG}^{Ser} (G37) is post-transcriptionally methylated by the Trm5 enzyme, yielding m¹G. The latter favors tRNA aminoacylation (here—leucylation) and may prevent frameshifts during the translation [17]. There are numerous visualization efforts to embrace mechanistic complexity and species-level diversity of three-layered genetic code. Notable examples of such efforts will be described below.

2.2 Visual Representations of the Genetic Code: Tables, Wheels, Hypercubes

The table of genetic code (Fig. 1) is one of the earliest visual representations of codon organization, as is the wheel of code (Fig. 3). Both depictions provide the reader with the information on an amino acid encoded by each codon. It is customary to present bases in the code as follows: U, C, A, G. Departures from this order lead to novel codon groupings, some of which suggest evolution of triplet code from a doublet one, and to the roles for sixfold degenerate codon boxes in balancing the GC content of prokaryotic genomes [22, 23]. Codons of genetic code can be arranged to illustrate other aspects of the process of gene expression. It is known, for example, that aRSs involved in tRNA charging fall into two structurally disparate classes I and II, and correspondence between aRS class and codons can be visualized in the form of hypercubes [24].

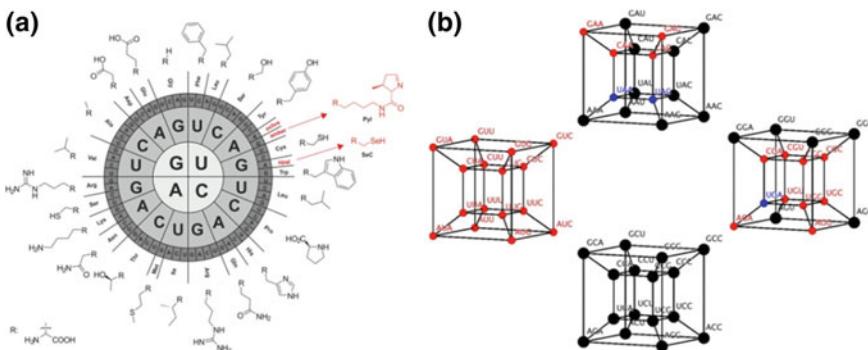


Fig. 3 Visual representation of genetic codes: different ways for different purposes. Textbook version in the form of wheel (a) provides information on the amino acid assigned to each codon. A triplet of mRNA ($5' \rightarrow 3'$; read from inside) is assigned to one of the 20 canonical amino acids or a stop codon. Chemical structures of amino acid side chains are shown on the outside. The natural expansion of the genetic code of selenocysteine (SeC) at opal and pyrrolysine (Pyl) at the amber and opal stop codons is depicted (shown in red). Codons can be arranged in four-dimensional cubes (b) according to their relatedness and type of amino acyl tRNA synthetase (class I, red; class II, black) used for their charging. Analogously, codons can be sorted out in n -dimensional graphs according to other properties of triplets or respective amino acids. Part a is from [25]; part b was adapted from [24]; <https://creativecommons.org/licenses/by/4.0/>

Further developments of the genetic code representation usually convert codons into a species-specific illustration of the strategies used to realize the code. For example, one may catalogue the entire set of anticodons present on tRNAs of a given species and pair it up with the table of genetic code (Fig. 4). This illustrates the structure of the tRNA pool used to decode mRNA in that species.

Hence, the universal genetic encoding paradigm becomes associated with a specific decoding strategy. One of the most information-rich views of the genetic code has been recently proposed by Grosjean and Westhof [27]. This is a circular representation, which depicts the genetic code, the strength of codon–anticodon interaction, the codon usage frequency and the modification status of certain nucleoside bases of tRNAs (Fig. 5). While the first two pieces of information are highly conserved across domains of life, the codon frequency and tRNA modifications are species-specific. Thus, for a given species one can depict a series of wheels of codes, which would differ in the tRNA nucleoside base for which post-transcriptional modification is summarized. This kind of visualization further extends our understanding of all the complexity of decoding strategies employed by a certain organism. Importantly, the authors described above wheel of code departed from a symmetrical arrangement of bases (compare with Fig. 3a) to re-assort the codons according to their thermodynamic properties. Such an integrated view of the genetic code better explains its function and evolution in the context of a species of interest. Much has to be done to propagate the use of such a sophisticated depiction of the genetic code throughout the biology. First of all, a comprehensive evidence-based description of tRNA

Codons				Anticodons			
UUU	UCU	UAU	UGU	GAA	GGA	GUA	GCA
UUC	UCC	UAC	UGC	GAA	GGA	GUA	GCA
UUA	UCA			U*AA	UGA		
UUG	UCG		UGG	U*AA, CAA	UGA, CGA		CCA
CUU	CCU	CAU	CGU	GAG	GGG	GUG	GCG
CUC	CCC	CAC	CGC	GAG	GGG	GUG	GCG
CUA	CCA	CAA	CGA	UAG	UGG	U*UG	UCG
CUG	CCG	CAG	CGG	UAG, CAG	UGG, CGG	U*UG, CUG	UCG, CCG
AUU	ACU	AAU	AGU	GAU	GGU	GUU	GCU
AUC	ACC	AAC	AGC	GAU	GGU	GUU	GCU
AUA	ACA	AAA	?	C*AU	UGU	U*UU	?
AUG, AUG	ACG	AAG	AGG	CAU, CAU	UGU, CGU	U*UU, CUU	CCU
GUU	GCU	GAU	GGU	GAC	GGC	GUC	GCC
GUC	GCC	GAC	GGC	GAC	GGC	GUC	GCC
GUА	GCA	GAA	GGА	UAC	UGC	U*UC	UCC
GUG	GCG	GAG	GGG	UAC, CAC	UGC, CGC	U*UC, CUC	UCC, CCC

Fig. 4 Side-by-side representation of codon table and corresponding anticodons within a set of tRNA expressed in a given taxon (archaea in this case). The asterisks indicate anticodons' first position post-transcriptional tRNA modifications necessary to unambiguously read the respective codon box. Blue shadow labels an example of a triplet whose decoding is enabled by post-transcriptional modification of the anticodon. Interrogation sign instead of AGA codon implies that it is still debatable what anticodon allows unambiguous reading of AGA. This figure was reproduced from [26], <https://creativecommons.org/licenses/by/4.0/>

modifications is currently available only for a few model organisms, limiting the usefulness of the proposed visualization to yeast, *E. coli*, human mitochondria and a few archaeabacteria. Many tRNA modifications are reversible and depend on the nutritional status of the cell [28–30]. By analogy to transcriptional regulation, a codon can be viewed as a *cis*-acting element of the regulatory device, whereas tRNA—as a *trans*-acting one. The interplay between these elements creates a continuum of regulatory opportunities for the cell to respond to internal and external signals via modulation of efficiency of codon reading. Current studies support the notion that adaptive translation is an essential feature of any biological system [31], yet it will be a challenge to predict and visualize this aspect of tRNA biology in current models.

To summarize, visualization of the genetic code is a vibrant research area. In recent years, the challenge has shifted from depiction of the encoding principle (which is nearly universal) to the incorporation of decoding strategies into the picture. This leads from the universal genetic code to species-level models. To fully embrace this approach, we still need to learn a lot about tRNA and mRNA biogenesis. Interestingly, after over 50 years of studies, genetic code visualization remains one of the least digitized aspects of molecular biology. This is because the paradigm of the universal genetic code leaves no room for automated analysis. Nowadays, the aforementioned efforts toward the inclusion of decoding details open the door for an automated analysis, for example through database mining for the information about codon frequency, types of tRNA modification enzymes and so on, and their conversion into human-readable diagrams. We think that the development of specialized

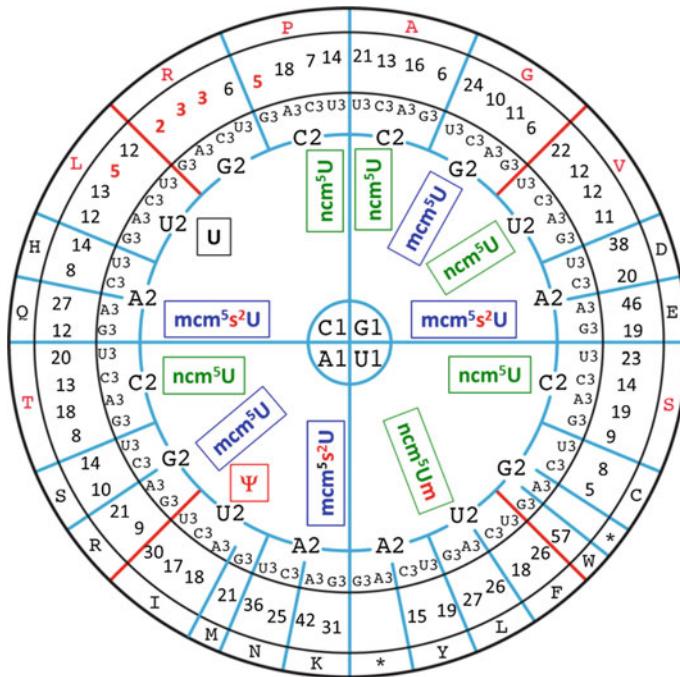


Fig. 5 *Saccharomyces cerevisiae* wheel of code with four layers of information. First, it represents codon identity. Codons are read from center to the edge of the wheel. Amino acids corresponding to un-split codon boxes are shown in red. Conventional one-letter code for amino acid is used; asterisk stands for stop codon. Second, it represents known post-transcriptional modifications (PTMs) of uridine (U34) in the first anticodon position of *S. cerevisiae* tRNAs. Types of different U34 modifications are shown in boxed rectangles (*mcm*⁵*s*²*U*, 5-methoxycarbonylmethyl-2-thiouridine; *mcm*⁵*U*, 5-methoxy-carbonylmethyluridine; *ncm*⁵*U*, 5-carbamoylmethyluridine; *ncm*⁵*Um*, 5-carbamoylmethyl-2'-O-methyluridine; ψ , pseudouridine). Third, it denotes the global codon usage for each codon of *S. cerevisiae* (counts per thousand). It is inserted between the circle for the third base and that for the amino acid type. Finally, four thick red lines divide genetic code into three sections according to Turner energy of anticodon–codon mini-helix formation. The bottom section consists of AT-rich codons having low Turner energy (weak codons); top section consists of GC-rich codons with strong mini-helix formation (strong codons). Intermediate values of free energy possess codons in the middle section. This is modified version of the figure S4Bc downloaded from <http://www-ibmc.u-strasbg.fr/spip-arn/spip.php?rubrique296&lang=fr> [27]

software for building such models would be of great help, for example, in case of industrially important organisms where optimization of gene expression is sought.

2.3 Modeling Genetic Codes: What Is Learned from Massive Codon Reassignments in Silico

The four-letter alphabet of the genetic code forms 64 codons; the latter encode 20 amino acids plus translation stop signal. Consequently, there are 21^{64} ($\sim 4 \times 10^{84}$) possible genetic codes. Additional constraints on theoretical genetic codes (e.g., a requirement to encode no less than 21 elements or specific assumptions about primordial code) reduce their number, albeit not significantly. Several mutually non-exclusive theories were put forward to explain the origin and evolution of extant genetic code. Current consensus view is that the code arose as a combination of a random event (“frozen” accident) that fixed certain proto-code, which further expanded and evolved to support vital cellular process of protein synthesis [10, 32]. Thus, natural genetic code is likely a result of an optimization process over evolutionary timescales, and so it should be fitter than random codes with regard to certain parameters. It remains an open question as to what goals the code is optimized for. Various aspects of protein synthesis, such as rate or accuracy of translation, are usually considered primary “suspects.” It is unlikely that the genetic code (and translation machinery in general) was shaped by a single requirement. Hence, the prevailing approach is to treat the evolution of genetic code as a multi-parameter optimization problem.

A number of studies addressed the optimality of natural genetic code by comparing it to alternative genetic codes. The immense number of the latter has prompted the researchers to impose certain constraints on what is considered a valid code. In one of the simplest scenarios, the natural and alternative codes would have the same number of codons per amino acid and the same impact of codon misreading. This was achieved by independent permutation of nucleotides in the first two codon positions and by allowing only an A↔G permutation in the third position (to satisfy the wobble rule stating that codons NNU/C cannot be distinguished by translation machinery). Therefore, a set of 1152 ($4! \times 4! \times 2$) alternative codes was generated and compared to the natural code with regard to two properties: an ability to encode arbitrary sequences of fixed length (n -mers; see Fig. 6) and the number of sense codons read by the ribosome frameshift translation error [33].

The probability to encode arbitrary n -mers is higher for natural genetic codes than in the vast majority of alternative codes. The longer the n -mer, the higher is the probability for natural code-based mRNA to carry such a sequence. This is thought to come from the fact that natural stop codons do not overlap with each other, yet they can arise from adjacent codons (within mRNA) for abundant amino acids through frameshifts. It is therefore possible that aforementioned property of real code arose as a side consequence of selection for minimization of effects of frameshift errors. Indeed, using natural genetic code, a ribosome would translate, on average, 15 codons out of frame prior to reaching the stop codon. In case of alternative codes, such a number is 23 codons. As compared to the real code, alternative ones would lead to measurable fitness defects because of longer and potentially more toxic protein products. The superiority of real code comes from the fact that its stop codons can

UGACA?				
frame 0	NN	UGA CAN N	$P_0 = 0$ (stop)	
frame -1	N N	UG ACA NN	$P_{-1} = P(NUG)*P(ACA)$	
frame +1	NN	U GAC ANN	$P_{+1} = P(NNU)*P(GAC)*P(ANN)$	
$P = (P_0 + P_{-1} + P_{+1})/3$				

Fig. 6 Coding sequence as a carrier of additional biological information: an example. 5-mer sequence UGACA, as a part of mRNA, can be read by ribosome, in triplets, in three possible frames (demarcated with green vertical lines between nucleotides). The probability of UGACA in the frame 0 (P_0) is zero, because it will lead to stop codon (boxed). Consequently, the probabilities of the 5-mer in frames -1 and +1 will be the product of probabilities of constituent triplets (the latter can be obtained from reference genomes, etc., and under different assumptions about codon usage frequencies). The described above scenario is for natural code; there will be other sets of stop codons for alternative codes

be easily “hidden” within a sequence. Therefore, the natural genetic code appears to display a robust behavior in the face of frameshift errors. This collaterally allows encrypting into mRNA extra information unrelated to the protein-coding capacity of the latter (signals for DNA binding proteins, microRNA, splicing sites, etc.) [34]. The aforementioned study indicates that the genetic code and amino acid abundances are coadapted. However, whether the genetic code is adapted to amino acid pools or vice versa still remains a point of debate [35]. Redundancy of the natural genetic code is the key reason for its capacity to carry additional layers of information.

It is possible to elaborate the other strategies for the judicious generation of sets of alternative genetic codes, which can be compared against the real one. However, even 10^9 codes would be a minuscule fraction of all theoretically possible variants. Such an approach cannot therefore guarantee that we understand properties of real genetic code on the basis of the analysis of the entire space of codes (or even a significant part of it). If more extensive search of this space would be undertaken, then the conclusions might prove incorrect. For example, natural genetic code appears to be adapted to minimize the adverse effects of the frameshifts arising through mutations and/or translational errors. One can think of natural code as “optimized” (or fit) with regard to these criteria as compared to over 1000 alternative codes. However, would a natural code still represent a global peak of fitness if we compare it against 10^9 codes? It is not a trivial question to address given the astronomical number of possible codes. There are no reliable analytic methods to search through vast spaces, and researchers often employ evolutionary algorithms to discover properties of genetic code. Briefly, the algorithm starts with initial population of codes, where random changes are introduced. Novel codes are evaluated on the basis of defined objective functions (e.g., fitness value for each code is determined). Fitter variants are taken to the next round of evolution and selection, and the procedure is iterated at researcher’s will or until no further improvement of fitness values is observed. This

approach has been recently implemented to assess how the optimality of genetic code is influenced in response to all possible single-point mutations that lead from one amino acid to another [36]. To deduce the fitness effects (costs) of such changes, authors have chosen eight different physicochemical properties (objectives) of amino acids (from AAindex database; <https://www.genome.jp/aaindex/>), such as isoelectric point, polarity, hydrophobicity and molecular weight. The algorithm started with 2800 random codes (each encoding 20 amino acids; stop codons were the same as in real code). The codes were permuted via genetic operator of mutation and evaluated, and a fraction of top-optimized codes were taken to the next round. The set of codes was replenished from the archive set to keep the size of population constant (2800). For each objective, the fitness of the code was calculated as the sum of squared differences between values of amino acid indices encoded by a pair of codons differing in one nucleotide:

$$F_i(\text{code}) = \sum_{(c_1, c_2) \in C} [p_i(c_1) - p_i(c_2)]^2 \quad (1)$$

where i is an objective index, C is the set of all pairs of codons differing in single nucleotide, c_1 and c_2 are codons, and $p_i(c_1)$ and $p_i(c_2)$ are the values of index i for amino acids encoded by c_1 and c_2 .

Each code is therefore represented by a vector of eight values, and the best codes are those minimizing the costs of amino acid replacements. Using Pareto evolutionary algorithm, authors went on to show that real genetic code is very close to theoretical best codes; e.g., it minimizes the costs of amino acid replacements as well as the latter. Nevertheless, there are codes more optimal than the real one with regard to as many as eight objectives combined. One of such theoretical codes is significantly different from the natural one: Only three codons (ACC, ACA and AGC) retained the same assignment in both codes, and codon boxes for Ser and Thr each consisted of 16 members. A general conclusion from evolutionary searches is that the natural genetic code does not represent even local optimum in the code space when a limited number of criteria are used to evaluate code optimality [37, 38]. An inevitable shortcoming of the conclusion is that the criteria used to compare the code are not necessarily those which the natural code was optimized for. Perhaps, the real code would achieve top optimality if more (or biologically more relevant) criteria are taken into account. It is possible to raise bacterial mutants that display level of accuracy of ribosomal protein synthesis higher than in the wild type, yet this accuracy comes at the expense of diminished growth rate [39, 40]. Likewise, current focus on the robustness or precision of the genetic code in face of mutations or translational errors might be misleading. This shortcoming notwithstanding, evolutionary algorithms show great promise in elucidation of the origin and properties of the genetic code. They allow sampling and iterative selection of codes across large swathes of the search space and might help uncover artificial variants with novel properties. Besides fundamental interest, the results of such studies can find their use in ongoing efforts to create orthogonal genetic systems [41, 42].

3 Estimating and Visualizing k -mer Occurrence in Genomic Sequences

The calculation of occurrence frequency of certain nucleotide subsequence (or k -mer) within a given sequence is among the most basic bioinformatic operations. It was the basis of the first ab initio gene-finding algorithms. For example, program FramePlot calculates the frequency of G/C nucleotides in the third position of the codons and thus recognizes open reading frames enriched with GC-ending codons, which are typical for GC-rich genomes of bacteria [43] (Fig. 7). Another key observation (made over 25 years ago and still valid today) is that in-frame hexamers are significantly underrepresented out of reading frame and vice versa [44]. This finding is at the heart of first versions of the gene recognition program GeneMark [45], suggesting that coding sequence features larger than codon are under selective constraints. Biased usage was demonstrated for tetranucleotides within and outside of the coding sequences [46]. Online applications were developed that calculate tetranucleotide frequencies to sort out metagenomic samples [47] and to perform whole-genome pairwise alignment [48]. Reasons for biased dinucleotide usage [49] are less well understood. Approaches based on k -mers are implemented in a number of detectors of errors in next-generation sequencing data [50, 51]. In this regard, the basic idea is that infrequent k -mers are the result of sequencing errors and thus library of short reads can be reduced to a set of correct sequences via k -mer indexing in the absence of reference genome. Finally, k -mer analysis constitutes an indispensable approach in genome-wide association studies—field of genetics that aims to identify standing genetic variation (mostly single-nucleotide polymorphisms) and associate it with certain phenotypic traits [52]. This non-exhaustive list of phenomena and applications built around them serves to remind of the importance of k -mer analysis for a

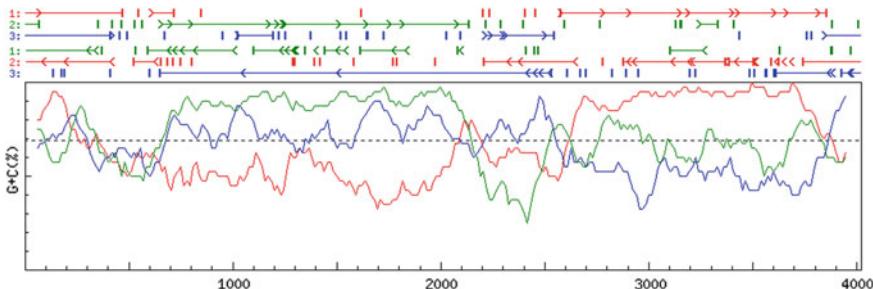


Fig. 7 FramePlot visualizes potential open reading frames within GC-rich nucleotide sequences. Here, a 4020-bp fragment of landomycin A biosynthetic gene cluster from *Streptomyces cyanogenus* S136 (accession #: AF080235) was used. The application calculates frequency of GC bases in third position of triplets in all six possible reading frames. Dashed line indicates overall GC content of the sequence being used (69%); colored lines indicate the % of GC in the third position of a triplet. Above the graph, symbols “l” and “>”, “<” indicate potential stop and start codons, respectively. The application was accessed from <http://www0.nih.go.jp/~jun/cgi-bin/frameplot.pl>

wide range of biological problems. Below, we proceed with a description of some of the available tools in this regard.

3.1 Overview of the Most Popular Tools

A number of k -mer calculators are available as desktop tools. Their extensive collection is present on OmicsTools Web site (<https://omictools.com/search?q=k-mer%20analysis>), and here we will review the most notable examples that deal with different aspects of the nucleotide sequence analysis. **Corseq** is the software to identify favored codons in RNA-seq data without the need for annotated genome sequence [53]. The software estimates transcript abundance by k -mer counting. Sequence element enrichment (**Seer**) and its Python-reimplemented version **pyseer** identify sequence elements (9–100 nt long) significantly enriched in certain phenotypes [54]. **Seer** allows alignment-free inferences of association of a certain SNP with a phenotypic trait, such as antibiotic resistance. **Jellyfish** and **Gerbil** permit fast, memory-efficient counting of k -mers, which can be visualized in the form of histograms. **KAnalyze** and **DSK** are fast k -mer counters that use low memory (especially the latter) and can be integrated into various sequence analysis pipelines. **Tallymer** uses enhanced suffix arrays to count k -mers of varying lengths. This method can be used to determine frequencies of various repeats in genomes. The **microTaboo** algorithm offers an efficient solution to the problem of finding unique (disjoint) k -mers (e.g., subsequences of length W that differ by more than k mismatches). This has a number of practical applications in the areas of SNP detection and selection of probes for diagnostic purposes, etc. In contrast to exact k -mer counters described above, **ntCard** estimates k -mer frequencies in genomic datasets from sample distribution. This is a preferred approach when researcher deals with high volumes (on a terabase scale) of sequence data [55]. **Squeakr** system is designed for either exact or approximate counting of k -mers. There is a number of benchmark studies which compare available k -mer counters in terms of speed, memory efficiency, scalability, etc., so that interested reader could choose the tool most suitable for the problem at hand [56].

3.2 Unmet Opportunities

All the progress and available options notwithstanding, there are ample opportunities for further improvement in the area of k -mer analysis. Poor visualization of obtained data is a common shortcoming of all available approaches, as the latter most often return results in tabular form. This is mostly explained by the fact that the development of new k -mer counters is focused on correction of sequencing errors. Positional information about k -mer is lost; namely, it is not known whether the k -mer is more frequent at the start or toward the end of the genes. To the best of our knowledge, at the

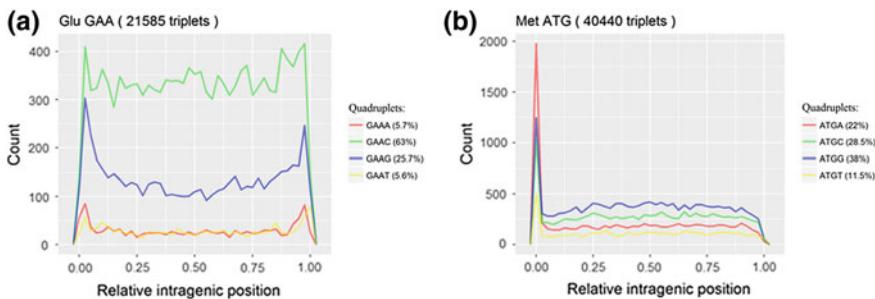


Fig. 8 Examples of observed quadruplet frequencies in *Streptomyces coelicolor* genome: biased associations for CIC, CIG, AIC (a) and biased ATGIA association at the beginning of the ORF (b). The x-axis represents cumulative distribution of the quadruplets over relative lengths (0.0–1.0) of all ORFs. The y-axis represents absolute count of any given quadruplet in the ORFome. Modified from [58]

moment there is no application to determine the frequency of k -mers on a genome-wide scale and represent the results in the form of cumulative plot, summarizing the inferred frequencies over the entire dataset (genome). This kind of analysis would be interesting to apply to certain genomes whose genes exhibit interesting and poorly understood codon and k -mer skews. For example, we showed previously that GC-rich genomes of Actinobacteria possess peculiar differences in usage of in-frame quadruplets across genes (Fig. 8); similar was observed for enterobacteria [57]. The development of simple online application to carry out such analysis and visualization would help better understand the distribution and significance of k -mer of different lengths in coding sequences.

4 Codon Indices

Due to the degeneracy of the genetic code, single amino acid sequence can, theoretically, be “spelled out” by an astronomical number of different codon sequences, of which a single one is actually used (Fig. 9). Each species prefers certain synonymous codons over the others to encode proteins. The non-uniform usage of synonymous codons is known as codon usage bias (CUB). In bacteria, CUB tracks with GC content of the genome [59] and tRNA abundance [60] or copy a number of tRNA genes [61]. The latter parameter also correlates positively with CUB observed in highly expressed genes in some animals [62]. CUB also influences mRNA and protein folding [63, 64]. These observations point to the possibility that in addition to neutral processes (such as mutation and genetic drift) CUB is a result of translational selection; namely, codon usage and tRNA pools are coadapted to increase the speed and/or accuracy of protein synthesis [65–67]. It is therefore necessary to have quantitative measures of CUB as they could help us understand mechanisms behind CUB and optimize coding sequences for different applications. A multitude of such measures

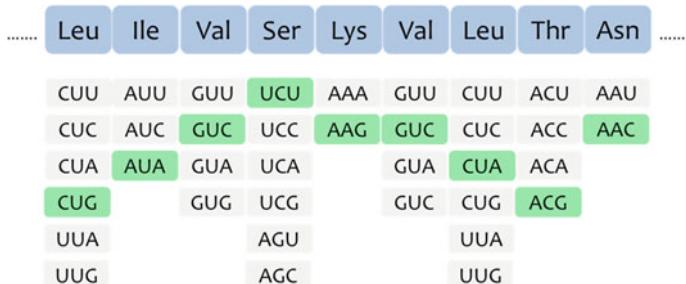


Fig. 9 An example of a nonapeptide sequence (blue boxes at the top): This sequence can be encoded by $6 \times 3 \times 4 \times 6 \times 2 \times 4 \times 6 \times 4 \times 2 = 165,888$ different synonymous codon sequences. One possible path through the space of codon sequences is highlighted with green boxes

have been proposed over the last three decades. Below, we will review those that are most frequently used or which were described recently and thus are not covered in the exhaustive 2012 review of CUB [1].

4.1 Definition and Diversity of Codon Bias Indices

The measures of CUB are known under a broad term of codon indices. The codon index employs a dedicated function to represent a certain (often quite narrow) aspect of codon usage with a single number. The codon index of a gene is a combined result of a contribution of constituent codons. Any amino acid encoded by more than one codon can exhibit bias. One-codon amino acids (Met and Trp) and stop codons are excluded from the analysis of CUB. The contribution of each codon to the index has to be carefully weighted to avoid the distortions caused, for example, by overrepresented amino acids. The quantification of CUB may pursue different goals. Several of the earliest indices were designed to reveal open reading frames (ORFs) of a nucleotide sequence. The rationale behind these approaches is that an ORF dominated by rare codons is unlikely to be protein-coding. Weakly expressed genes carry fewer rare codons than can be expected from background nucleotide frequencies [68]. Likewise, codon metrics can be employed to correct sequencing errors and filter out random ORFs. As different species exhibit a different CUB, the indices can be used to detect genomic regions that arose by horizontal gene transfer [69]. Conversely, if some genes or gene sets across different species show similar levels of codon sequence conservation (as described by a certain index), then this can be construed as evidence for coevolution of abundance of functionally related proteins. Perhaps, the most frequent use of codon indices is for prediction or visualization of protein expression level. Highly expressed prokaryotic genes can be deduced on the basis of calculation of codon indices, and there are extensive databases where such information is collected for some well-studied bacteria [70]. Different indices can be classified with regard to the force that the former illuminate. Some indices,

such as the effective number of codons, report on combined effects of mutation and selection. Many codons specifically describe translational selection on codon usage. Methodologically, most of codon indices fall into one of the two categories. The first category calculates the deviation of codon usage of gene of interest from the expected codon distribution (e.g., calculated from nucleotide frequencies). More often, indices compare the codon usage of a gene to the codon usage of a reference set. Still, some indices do not fit this dichotomy, while others can be used to study codon usage against either the expected distribution or specific sets of reference genes. Essentially, the “cottage industry” of codon indices is based on the exploration of virtually endless ways to select experimental datasets and strategies of their comparative analysis. The ever-blurring boundaries between different approaches toward quantifying CUB lead us to abandon simplistic classification. The peculiar features of each index and alternative ways of its implementation (if available) will be discussed shortly. We adhere to the scheme described in Chap. 13 of [1] for the notation of indices in this section.

4.2 Biological Significance of Several Popular Codon Indices

4.2.1 Relative Synonymous Codon Usage (RSCU)

This codon metric requires at least two sets of sequenced genes significantly differing in expression; further, we refer to them as highly and lowly expressed genes. Then, RSCU is determined from codon frequencies for each synonymous codon of each amino acid:

$$r_{ac} = \frac{o_{ac}}{\frac{1}{k_a} \sum_{c \in c_a} o_{ac}} \quad (2)$$

where o_{ac} is count of codon c for amino acid a in a gene or set of genes; k_a is the number of synonymous codons. The RSCU values are the ratio of observed number of a codon to the expected one, when it occurs by chance. $RSCU = 1$ when there is no synonymous codon usage bias; $RSCU > 1$ when codon usage is more frequent than average for the amino acid.

As a reference set, in their original 1986 work Sharp and coworkers have chosen genes known to be highly expressed. Recently, Paulet et al. [71] proposed to use ribosome profiling data instead of codon counts in calculating RSCU. The ribosome profiling, or Ribo-seq, determines how many times ribosome occupies every codon of transcript. Thus, here one deals with counts of ribosome occupancy on different codons, which is a measure of mRNA translation level. Authors dubbed their version of RSCU as RSCU_{RS} and went on to show that both aforementioned indices strongly correlate ($r > 0.97$) for a number of model species, such as worm (*Caenorhabditis elegans*) and yeast (*C. albicans*). Authors calculated index values for a central segment of transcripts, from 20th to 200th codon. Differences between RSCU and

RSCU_{RS} become more apparent when shorter ranges were used (20–50; 20–100). This is a common property of all codon indices: Some minimal number of codons is necessary (80–100) to deduce the reliable information. There are a number of advantages in the use of Ribo-seq data. One can directly assess the significance of CUB for translation; highly translated genes can be automatically selected even if they are not functionally annotated; most importantly, there is a possibility to assess the influence of different conditions (nutrition, tissue-specific aspects and so on) on translation and, therefore, on RSCU_{RS}. Indeed, it has been already reported that in *E. coli* amino acid starvation leads to reduced translation of abundant codons from sixfold degenerate families (e.g., leucyl codons CUU, CUA, CUC), while the reading of rare codons (UUR) and CUG still remains robust [28]. Hence, RSCU_{RS} should, in principle, be able to capture the dynamic nature of the efficiency of mRNA decoding—an aspect of biology not assessable with the other indices. An overall consensus is that highly translated genes are enriched with certain (“optimal”) codons; this seems to agree with the original work of Sharp and Li [72].

4.2.2 Codon Robustness Index (CRI)

The empirical metric CRI was deduced as a result of aforementioned study [28] on codon degeneracy lifting in the face of environmental perturbations. The basic idea of this measure was articulated in some previous works, showing that favorite codons are not those that are most abundant in the genome, but rather those which cognate tRNAs are most efficiently charged under amino acid starvation [73, 74]. The CRI describes the robustness of a protein synthesis rate to a limited amount of a certain amino acid. The competition between tRNA isoacceptors for aminoacylation is proposed to be the key determinant of the hierarchy of mRNA translation rates during amino acid starvation. The index is computed by summing over log weights of codons ω_c that belong to a subset of codons corresponding to the limited amino acid ($c \in C_{a_lim}$):

$$\text{CRI} = \sum_{c \in C_{a_lim}} \log_2 \omega_c \quad (3)$$

The ω_c values for Leu and Arg were calculated based on the robustness of synthesis of a panel of yellow fluorescent proteins (YFPs) whose coding sequences carried different combinations of synonymous Leu and Arg codons. If, for example, YFP coding sequence carries 22 Leu codons (7 CTA and 15 CTG) and taking that $\log_2 \omega_c = W_c$, one deduces that synthesis rate of given YFP variant will be $7W_{CTA} + 15W_{CTG}$. The above line of reasoning was applied to find ω_c . The ω_c values for Leu codons were as follows: CTG, 1; TTG, 0.91; TTA, 0.88; CTC, 0.67; CTT, 0.61; and CTA, 0.45. The ω_c values for codons other than those cognate to the limiting amino acid were set to 1. Authors calculated Z-scores for CRI to estimate the deviation of observed CRI values from the expected ones. The latter were estimated on a dataset of 4.3 million random

sequences (each of 4300 ORFs in the *E. coli* genome was permuted 1000 times). Using 92 ORF-YFP translational fusions, authors demonstrated that among several indices of translational efficiency, such as CAI and tAI (see below), CRI shows the highest correlation with experimental data. Although CRI appears to model quite a narrow biological scenario, natural ecosystems are heavily dominated by nutrient-limited conditions; human gut [75, 76] and soil [77, 78] are two well-known examples of such ecosystems.

4.2.3 Relative Adaptiveness

The RSCU represents one of the solutions to normalization problem in calculating codon indices. Another way of normalizing data is to express the frequency of each codon with regard to the count of the most frequent codon in the dataset ($\max o_{ac}$). This idea is embodied in relative adaptiveness metric ω_{ac} :

$$\omega_{ac} = \frac{o_{ac}}{\max_{c \in C_a} o_{ac}} \quad (4)$$

Here, the most frequent codon will have $\omega_{ac} = 1$, while the ω_{ac} of all the other codons will be <1 .

4.2.4 Frequency of Optimal Codons (Fop)

Fop is the ratio of the number of optimal codons within a gene (dataset) to a total number of synonymous codons. This is perhaps the earliest codon measure and the first one which relies on reference information. The codon optimality can be derived from chemical properties of nucleotides, or genome parameters, such as GC content, CUB or copy number of tRNA genes. The latter seems to be the most important one, for reasons mentioned at the beginning of this section. It permits to define translationally optimal codons as those for which there is the highest tRNA gene copy number. Having defined a subset of optimal codons o_{opt} , one calculates Fop:

$$Fop = \frac{o_{opt}}{o_{tot}} \quad (5)$$

where o_{tot} is the total number of a codon in the analyzed sequence.

4.2.5 Codon Preference (P)

This is another reference-based measure that was used to locate protein-coding genes and detect frameshift sequencing errors. One first calculates the likelihood ratio ω_{ac}^P :

$$\omega_{ac}^P = \frac{f_{ac}}{e_{ac}} \quad (6)$$

where f_{ac} is the observed frequency of codon c for amino acid a in the gene, and e_{ac} is the frequency of that codon expected from background nucleotide frequencies in the genome (computed as a product of nucleotide frequencies at the three codon positions: $e_{ac} = b_1 b_2 b_3$).

To determine P for a gene, the above calculated values (in the form of log-likelihoods) are summed over the entire length of the gene L :

$$P = \exp\left(\frac{1}{L} \sum_{i=1}^L \log \omega_{ac}^P(i)\right) \quad (7)$$

4.2.6 Codon Adaptation Index (CAI)

This is the most popular CUB metric that compares the codon usage in a gene of interest against the codon usage in highly expressed genes [72]. The latter can be derived from experimental data; in their absence, ribosomal protein genes can be used [79], as they constitute one of the most highly expressed groups of genes across all domains of life. Alternatively, highly expressed genes can be inferred from prevalent CUB for a given genome in course of an iterative procedure outlined in [80]. For an mRNA consisting of L codons, the CAI can be computed as follows:

$$\text{CAI}_{\text{mRNA}} = \frac{1}{L} \sum_{a \in A} \sum_{c \in C_a} o_{ac} \ln(\omega_{ac}^{\text{ref}}) \quad (8)$$

where C_a stands for codons encoding amino acid a; A is set of all amino acids a; o_{ac} is the observed count of codon c for amino acid a in a given mRNA; ω_{ac} is the relative adaptiveness of codon c encoding a in a given mRNA (computed according to Eq. 3); the reference relative adaptiveness is calculated on the basis of either mRNAs or ribosomal genes or other chosen datasets.

Reference genes exhibit the highest $\text{CAI} = 1$, which will be ≤ 1 for other genes. Several improvements of CAI were proposed that deal with different mutational biases, gene expression level, translation efficiency of synonymous R- or Y-ending codons or identification of highly expressed gene set [81]. On the basis of these considerations, a novel index of translation elongation (I_{TE}) has been proposed that is computed similarly to CAI [82]. The I_{TE} was applied in the re-analysis of expression data from a library of 154 synonymous genes encoding green fluorescent protein [83], pointing to the possible explanation as to why translation elongation was under-estimated in the original study as the contributor to protein expression. Recently, two corrections to CUB indices, including CAI, were proposed that take into account the fact that in all domains of life many mRNAs carry sequences that induce -1 ribosomal

frameshifting [84]. Such events incur costs to translational efficiency that may not be reflected by traditional CUB measures. On the basis of several datasets, correction for -1 frameshifts was shown to lead to a higher correlation between CUB and expression level.

4.2.7 tRNA Adaptation Index (tAI)

The tAI is a measure of adaptation of codon sequence to cellular tRNA pool. Like in CAI, tAI is an average value of each codon adaptiveness. The latter is computed in two steps. First, absolute adaptation W_c for codon c is found:

$$W_c = \sum_t (1 - s_{ct}) T_{ct} \quad (9)$$

where s_{ct} is efficiency codon–anticodon coupling (has to assign separately); T_{ct} is the number of tRNA molecules t that recognize codon c (estimated from tRNA gene copy numbers).

The relative adaptiveness is then computed as a ratio of absolute adaptiveness to the maximum W_c value of given amino acid:

$$\omega_{ac}^{\text{tAI}} = \frac{W_{ac}}{\max_{c \in C_a} W_{ac}} \quad (10)$$

The mean ω_{mean} is used as relative adaptiveness if W_c is zero. Finally, tAI of a gene is found from all relative adaptiveness values of individual codons:

$$\text{tAI} = \exp\left(\frac{1}{o_{\text{tot}}} \sum_{c \in C} o_c \log \omega_c^{\text{tAI}}\right) \quad (11)$$

The tAI hinges on the fact that different tRNAs recognize the codon with different affinities, due to wobble interactions and different pools of anticodons available for each codon. One needs the following pieces of information to compute this index: the list of anticodons that recognize codons; complete and correct catalogue of tRNA genes for a given genome; reference set of highly expressed genes; and efficiency of codon–anticodon recognition (wobble interaction weights s_{ct}). As mentioned above, tRNA gene copy number serves as a proxy to tRNA abundance in the original implementation of tAI. Recent technological advances allow a direct quantification of tRNA from RNA-seq data, offering more precise estimation of tAI [85]. A version of tAI, called normalized translational efficiency nTE, was proposed in 2013 that takes into account both tRNA abundance and demand for it; the latter is inferred from the codon usage [86]. For a long time, s_{ct} was derived from gene expression data in *S. cerevisiae*. In 2014, a generic approach has been proposed for species-specific estimation of the weights without resorting to gene expression data

[87]. The approach capitalizes on the assumption that highly expressed genes should have both higher tAI (more adapted to tRNA pool) and higher CUB (less uniform distribution of synonymous codons). There must be, consequently, a significant correlation between CUB and tAI. One may therefore find species-specific values of s_{ct} that optimize the correlation between the aforementioned metrics. Authors used a derivative of relative codon usage bias index, RCB [88], as a measure of CUB; its chief advantage is that it is dependent only on coding sequence.

4.2.8 Synonymous Codon Usage Order (SCUO)

This is information theory-based measure of CUB. The entropy for each amino acid is computed through the normalized difference between maximum and observed entropies:

$$E_a = \frac{\max(H_a) - H_a}{\max(H_a)} = \frac{\log_2 k_a - H_a}{\log_2 k_a} \quad (12)$$

where k_a is the number of synonymous codons encoding amino acid a.

Then, SCUO can be computed as follows:

$$\text{SCUO} = \sum_{a \in A} F_a E_a \quad (13)$$

where F_a is relative frequency of amino acid a in the sequence.

4.2.9 Effective Number of Codons (Nc)

Nc characterizes the total number of different codons in a given gene sequence [89]. Therefore, Nc ranges from 20 (each amino acid of the protein is encoded by a single codon) to 61 in case each synonymous codon is present in the sequence. Nc estimates the departure of the observed codon usage from the expected one (assuming the uniform usage of synonymous codons as a null distribution). Relative codon frequencies f_{ac} from counts of codon c for amino acid a are computed:

$$f_{ac} = \frac{o_{ac}}{\sum_{c \in C_a} o_{ac}} \quad (14)$$

Homozygosity Z_a of an amino acid a is then computed, implying that different synonymous codons are treated as different alleles of the gene:

$$Z_a = \frac{o_a \sum_{c \in C_a} f_{ac}^2 - 1}{o_a - 1} \quad (15)$$

The effective number of codons N_a for amino acid a is found:

$$N_a = Z_a^{-1} \quad (16)$$

N_a takes values from 1 to the number of synonymous codons k_a for amino acid a. The effective number of codons N_c for a gene will be the sum of average Z_a for different redundancy classes k out of the entire set K of all such classes:

$$N_c = \sum_{k \in K} n_k \bar{N}_{a=k} \quad (17)$$

where

$$\bar{N}_{a=k} = \frac{1}{n_k} \sum_{a \in k} N_a \quad (18)$$

Special rules regulate the calculation of N_c in cases, when codon usage of a gene is more uniform than expected or when certain amino acids (and respective codons) are missing in the protein product [1]. For example, if one or a few amino acids within class k exhibit(s) strong CUB, while the others in the same class do not, then N_c for a gene could be calculated as the sum of individual amino acids (and not the averages over k) [90]:

$$N_c = \sum_{a \in A} N_a \quad (19)$$

A few improvements of N_c were proposed that take into account GC content of the coding sequences [91–93].

4.2.10 Information-Based Codon Usage Bias (iCUB)

This is another metric of CUB based on Shannon's information entropy [94]. The entropy H_a for an amino acid a encoded by a set of synonymous codons C_a from degeneracy class k will be

$$H_a = - \sum_{c \in C_a}^{k_a} f_{ac} \log_2 f_{ac} \quad (20)$$

where f_{ac} is frequency of encoding amino acid a with codon c. H_a for all amino acids can be converted into information entropy of the entire gene as follows:

$$H_g = \sum_{a \in A}^{20} p(a) H_a \quad (21)$$

Maximum possible information entropy H_m of the gene is required in order to derive useful information from H_g . The H_m can be inferred in various ways and may include additional constraints, such as GC content. Under conditions of no extra information and constraints, the H_m can be computed as follows:

$$H_m = \sum_{a \in A}^{20} p(a) \log_2 k_a \quad (22)$$

The H_m values computed in this way may not represent a null hypothesis about properties of the gene under study: Its codon structure is not random as it is under selection constraints to carry out a biological function. Therefore, it is perhaps more logical to infer H_m from random sequences that match the GC and amino acid contents of the gene, according to procedure outlined in [94]. Then, the raw CUB score is:

$$S_g = \frac{H_g}{H_m} \quad (23)$$

The iCUB values will range from 20 to 61, similar to Nc:

$$\text{iCUB} = 20 + S_g(61 - 20) \quad (24)$$

4.2.11 Codon Deviation Coefficient (CDC)

This measure compares observed CUB against expected random distribution taking into account both GC and purine contents. The cosine distance metric is adopted to measure the difference, and bootstrapping is used to assess the statistical significance of the results [95]. The CDC value will be within 0–1 range. The codon position-specific nucleotide contents are computed as follows:

$$A_i = (1 - S_i)R_i, T_i = (1 - S_i)(1 - R_i), G_i = S_i R_i, C_i = S_i(1 - R_i) \quad (25)$$

where A_i , T_i , G_i , C_i are standard nucleotide frequencies, and S_i and R_i stand for GC and purine content at the i th codon position ($i = 1, 2, 3$). Then, the expected usage of any sense codon e_c is the product of expected nucleotide frequencies at three positions ($b_1 b_2 b_3$) divided by the sum over all sense codons:

$$e_c = \frac{b_1 b_2 b_3}{\sum_{c \in C} s_c b_1 b_2 b_3} \quad (26)$$

$$\text{where } s_c = \begin{cases} 1, & \text{if } c \text{ is sense codon} \\ 0, & \text{otherwise} \end{cases}.$$

When both expected (e_c) and observed (f_c) codon usages are determined, CDC computes the distance coefficient for a given gene:

$$\text{CDC} = 1 - \frac{\sum_{c \in C} e_c f_c}{\sqrt{\sum_{c \in C} e_c^2 \sum_{c \in C} f_c^2}} \quad (27)$$

Statistical significance is tested by bootstrapping the analyzed sequence to generate replicates with matching length and nucleotide, S and R contents. Statistically significant CDC is one having two-sided bootstrap p -value < 0.05 . CDC is a pioneering example of the use of bootstrap resampling for statistical analysis of CUB. Using this index, the authors revealed interesting differences in CUB of genes coding for several ribosomal proteins. These could be interpreted as indicative of differences in evolutionary processes that remained undetected by the other CUB measures.

4.2.12 Nonsense Error Adaptation Index (NAI)

While the other described above indices are heuristic or statistical at their core, NAI is built upon a mechanistic model of protein translation. The coding sequence is linked to its protein production costs. NAI measures the extent to which a coding sequence is adapted to minimize the cost of nonsense errors during translation, as compared to the distribution of all possible synonymous variants of the gene (alleles). Starting from a set of parameters (such as protein synthesis rate and mutation rate), one follows a four-step procedure to compute NAI, as detailed in [96]. First, per-codon elongation and nonsense error probabilities are calculated. Second, protein production cost in the face of nonsense errors (η_{obs}) is calculated. In the third step, there are calculated central moments of η_{exp} (and variance thereof) across the entire space of synonymous variants (approximated by some continuous distributions) of the given coding sequence. In the fourth step, NAI is computed:

$$\text{NAI} = -\frac{\eta'_{\text{obs}} - \eta'_{\text{exp}}}{\sqrt{\text{Var}(\eta')}} \quad (28)$$

where notation ' means that values were Box-Cox-transformed to more precisely reflect the size of genotype spaces with less adapted alleles.

Using *S. cerevisiae* as a test case, the authors showed that over 90% of all coding sequences in its genome are more adapted to reduce the burden of nonsense errors than would be expected by chance. NAI values increased with the codon position within the genes, although they were also higher at the gene start than expected. The latter observation may indicate that the start of a coding region exhibits tendency to evolve toward the higher translational efficiency to avoid ribosome collisions at the stage of translation initiation [97, 98]. The explicit mechanistic principle of the development of nonsense error cost index described in this work can be extended to the development of similar indices focused on translational accuracy or efficiency.

4.2.13 Ribosome Overhead Cost Stochastic Evolutionary Model of Protein Production Rate (ROC SEMPPR)

This is a mechanistically interpretable Bayesian model that infers mutational biases, translational efficiencies and gene expression estimates on the basis of genomic sequence alone [99]. The ROC SEMPPR is a further development of the protein production cost (η) concept outlined above. Here, highly expressed genes should be those that exhibit the strongest CUB to reduce ribosome pausing and errors. Under the conditions of large effective population size, where these genes should show signal of adaptation, the pattern of CUB indicates protein production rate Φ for the gene. For lowly expressed genes, the CUB is largely explained by mutation biases. Hence, ROC SEMPPR is a mechanistic codon translation model defined in the population genetics framework, which allows to generate a posterior probability distribution for Φ . The authors point out that this model may not provide reliable estimates for organisms whose population sizes are either too small or too large. The model comes with a number of other necessary simplifications, such as an assumption that there is no interaction between ribosomes or between ribosomes and an RNA polymerase. Both simplifications may not reflect the biological reality for some taxa, such as bacteria, where coupled transcription–translation is commonplace [100]. A further improvement of the model is possible by taking into account protein translation factors and coding sequence properties other than used in the original study.

4.2.14 Logarithm of the Odds of Expression Score (θ)

This metric was derived from large-scale expression of genes of diverse phylogenetic origin in *E. coli* under the control of T7 RNA polymerase [101]. Protein expression level for each protein was scored from experimental data (SDS-PAGE) on an integer scale from none (0) to the highest (5). Then, multi-parameter binary logistic regression modeling was employed to reveal the contributions of different mRNA sequence parameters to the expression of proteins that fall into different expression “bins” (0–5). Particularly, the analysis showed that frequency of certain synonymous codons is higher in different segments of coding sequence (see below) and differs for proteins from different “bins,” leading to codon slopes—a relationship between codon frequency and its influence on expression level. After filtering out non-contributing and redundant factors, a logarithm of the odds of observing 5 versus 0 expression score for a gene was proposed:

$$\begin{aligned} \theta = & 2.0 + 0.054\Delta G_{\text{UH}} - 1.6I + 6.6a_{\text{H}} - 6.3a_{\text{H}}^2 - 1.8g_{\text{H}}^2 + 0.80u_{3\text{H}} \\ & + 0.86 \sum_{c \in C} \beta_c f_c + 0.078s_{7-16} + 0.063s_{17-32} - 1.8d_{\text{AUA}} - 16r \\ & - 0.0012L - \frac{520}{L} \end{aligned} \quad (29)$$

where ΔG_{UH} is the predicted energy of free folding of the 5'-untranslated terminal region plus first 16 codons of a gene (“head”), as predicted by the RNAstructure software [102]; I is a binary indicator only if $\Delta G_{\text{UH}} < -39 \text{ kcal mol}^{-1}$ and GC content of codons 2–6 is $> 62\%$; a_{H} and g_{H} are A and G frequencies in codons 2–6; $u_{3\text{H}}$ is the U frequency at the third position in codons 2–6; β_{c} and f_{c} are the slopes and frequencies of each non-termination codon; s_{7-16} and s_{17-32} are the mean slopes for codons 7–16 and 17–32 (a function of their frequency in proteins from the highest expression bin, as defined in this study); d_{AUA} is a binary variable that is 1 only if there is a least one AUA-AUA dicodon; r is the amino acid repetition rate (defined as distance at every position in the sequence to the next occurrence of the same amino acid, toward the 3' end of the gene); L is the coding sequence length.

The empirical model derived from the coding sequences and their expression data emphasizes the differential contributions of the head of the coding sequence and the rest of the gene, or the tail (approximately after codon 32) to the protein expression score. The folding of the head is very influential, and the individual input of each head codon is 3 times stronger than that of tail codons. However, as the tail is longer than the head, the codons of the former are in total ~5 times more influential than the head. Hence, the model underlines the unique requirements of the head for translation initiation; there must be optimal codon structure to minimize folding stability and ensure the accessibility for ribosome docking. The codon usage at the head was linked to mRNA folding in several other recent studies [103–105]. In the tail, the codon structure impacts the elongation efficiency and mRNA stability. The mechanisms of the latter phenomenon remain obscure [106]; post-transcriptional mRNA modification might be one of the factors [107]. Another hypothesis is that the enrichment of the transcript with rare or non-optimal codons slows translation and might elicit degradation mechanisms similar to that of nonsense-mediated decay in eukaryotes [108–110]. There is weak (if any) correlation between θ metric and known codon indices, such as CAI and tAI, as well as estimated cognate tRNA concentrations. Also, non-optimality of some of the rare codons (as deduced in previous works) was not confirmed in this study. On the basis of the model, two methods were designed to optimize coding sequences for top expression in *E. coli*. The “six amino acid” (6AA) method replaces all Arg, Asp, Glu, Gln, His and Ile codons with “optimal” synonyms—CGT, GAT, CAA, GAA, CAT and ATT, respectively. All of them carry A/U in the third codon position. The “31 codon folding optimization” (31C-FO) method manipulates the 31 head codons [102] to maximize ΔG_{UH} and to keep folding energy of the tail near $-10 \text{ kcal mol}^{-1}$. The coding sequence optimization led to a consistent increase in protein expression, irrespective of the method, provided that both head and tail were optimized. This increase was due to the higher translation efficiency and decreased degradation of the mRNAs.

4.2.15 Mean Codon Stabilization Coefficient (CSCg)

This is a codon metric which links the codon occurrence to the mRNA half-life. Several transcriptomic and proteomic studies (see above) indicated that enrichment

of coding sequence with optimal codons (e.g., those that are efficiently translated) correlates with increased mRNA stability. Codon stabilization coefficient for *S. cerevisiae* has been determined for each codon from experimental data [111], and the latter can be used to compute CSC for each yeast gene (CSCg):

$$\text{CSC}_g = \sum_c \left(\text{CSC}_c \frac{o_c}{L} \right) \quad (30)$$

where CSC_c is CSC value for a given codon c; o_c is count of codon c in gene g; L is the length of gene g in codons. To ascertain the biological meaning of CSCg values observed in real genes, these were compared against values computed from shuffled genomes [112].

4.3 Online Applications and Databases to Calculate and Visualize Codon Indices

Some of the earliest indices are already part of bioinformatics suites (such as EMBOSS) and biological sequence analysis software packages. For example, MEGA X [113] supports the calculation of RSCU, and DAMBE7 [114] allows computing RSCU and several versions of CAI. INCA [115] is a software package fully devoted to codon analysis. It computes common indices such as CAI, Nc and codon bias and represents them as either text or plots, as shown in Fig. 10. Program CodonW [116] calculates the following indices: RSCU, CAI, Nc, CBI and Fop. The first three indices from the aforementioned list can also be computed with ACUA toolkit [117]. CAIJava is Java-based program that calculates CAI of a gene of interest using as a reference a set of genes with the highest CAI scores (<http://www.ihes.fr/~carbone/materials/description.html>). The reference set is identified through an iterative procedure of selection of coding sequences harboring very frequent codons [80]. The latter feature distinguishes CAIJava from many other CAI calculators which take known or presumably highly expressed genes (usually for ribosomal proteins) or codon usage table for the organism as a reference. Program CodonO computes SCUO index and visualizes data in the form of tables and plots [118]. xtRamp is a software package that identifies so-called ramp sequences from protein-coding genes [119]. The ramp sequence is a 5'-terminal segment of the gene adjacent to the start codon which is preferentially populated by slowly translated codons. This is thought to ensure a more uniform placement of ribosomes at the translation initiation stage, thus avoiding their collisions [120]. The software makes use of tAI and relative codon adaptiveness to identify the ramp sequences.

A number of online tools support CUB analysis. Being perhaps the most popular index, CAI enjoys the widest variety of tools for its calculation—see [121–123] and <https://www.biologicscorp.com/tools/CAICalculator/#>. XKRB1khS_IU; <http://www.bioinformatics.nl/cgi-bin/emboss/cai>. GCUA [124] compares codon and

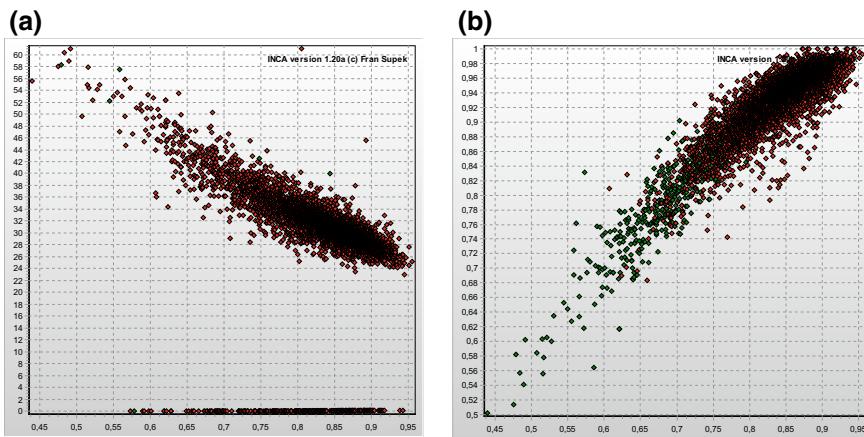


Fig. 10 Visualizing CUB on a genome-wide scale with INCA. The entire set of *Streptomyces coelicolor* A3(2) coding sequences was loaded into the program, which calculates a number of genomic and codon usage parameters. Here, the plots depict correlation between CAI (x-axis; 0–1) and Nc (y-axis; 0–61) in part **a**, and between CAI and GC content in third codon position in part **b**, for all *S. coelicolor* genes. Genes on minus and plus strands of the chromosome are shown as red and green dots, respectively. By pointing to each dot, one can get information about protein product of the gene and correlation coefficient between CAI and Nc. The figure was generated with INCA version 1.20

amino acid usage of query sequence against codon usage table for the source organism and presents the result as bar graphs. The stAIcalc [125] is an online tAI calculator which features pre-computed wobble interaction weights for 100 organisms. The output file contains tAI values in a tabular form. The calculator can be accessed at <http://tau-tai.azurewebsites.net/>. Stand-alone version of the program permits to optimize the weights for new species.

Several databases contain pre-computed indices for selected organisms. Kazusa database (<https://www.kazusa.or.jp/codon/>) is an open collection of codon usage tables for about 36,000 organisms. The codon usage tables provide information (in a tabular form) about the absolute numbers of each codon in a genome and codon frequency per 1000 codons. However, this database is rather outdated as it is based on GenBank file released in 2007. HIVE-CUT is another online storage for codon usage tables of all (approximately 850,000 as of April 2019) genomes deposited into GenBank [126]. HIVE-CUT requires registration and is updated every two months. This database also contains pre-computed Nc indices for all coding sequences. STADIUM contains species-specific tAI values pre-calculated for 148 species [127]. Besides raw data in tabular form, one can find the distribution of tAI values across different biological pathways.

4.4 Future Directions

The codon usage indices have been used in a vast number of recent studies and become an indispensable tool for the analysis of genomic sequences [128]. Extensive body of data shows that, largely, there is a high correlation between the gene's CUB, abundance of cognate tRNAs, mRNA and protein levels [129–133]. Codon optimization or de-optimization has become a routine biotechnological tool to boost or suppress, respectively, protein expression levels in prokaryotic platforms [101, 134, 135, 136]. This suggests that translational selection is an important driver of CUB. However, it is still a formidable challenge to disentangle the contributions of neutral and adaptive processes to the final codon structure of the genome [137, 138]. Rules, causality, benefits and risks of codon optimization remain debatable for mammalian and plant systems [139–142]. The protein expression level is inevitably a dynamic parameter, a net result of multitude of biological factors [143]. There is an inherent tension between static estimation of expression level portrayed by a given codon index and expression level ranges revealed with omics approaches. This divide is not insurmountable as researchers continue to explore novel ways to incorporate biological variables into CUB metrics; some examples have been mentioned in preceding section. Besides the aforementioned conceptual issue, the understanding and applicability of CUB indices suffer from unresolved methodological shortcomings. First of all, there is no single platform where all or the majority of (most popular) CUB indices could be computed for a given gene or genome. Such a platform would facilitate the comparative analysis of performance of these indices and possible confounding effects of the biases other than translational selection (gene length, GC content, etc.). Indeed, a recent benchmarking study of several indices based on measures from random expectation (Nc, SCUO, RCBS, CDC) revealed that different indices produce large heterogeneity of results when challenged with the same synthetic datasets [94]. There is a broad range of correlation values between indices thought to gauge the same aspect of codon function. Hence, either these indices are sensitive to unanticipated biases, or they measure different aspects of codon usage. More extensive benchmark studies are needed to more fully understand the CUB index behavior. A corollary to this is the need for more rigorous statistical validation of CUB indices. Currently, only CDC index comes with a means for estimation of its statistical significance [95].

The computed codon indices are most often represented in a tabular form. Program package INCA offers perhaps the richest menu of options for CUB visualization. Yet, for many indices (e.g., CDC and iCUB) there are still no online applications, greatly limiting their accessibility, especially to researchers non-conversant with command-line program interfaces. There is ample room for an improved representation of popular indices as well. For example, a graphical representation of per-codon CUB index values across the entire coding sequence is often useful [144], yet this feature is not implemented in any visualization tool.

5 Codon Context

The first observations that codons tend to be specific to a certain preferred neighborhood, or context, which influences the mRNA function, date back to the 1980s [44, 145, 146]. Today, the phenomenon of codon context is a solid line of investigations, as judged from over 1500 entries in PubMed mentioning this term. The codon context comes in a great variety of forms and functional consequences. The genetic code degeneracy largely enables codon context, as different nucleotide sequences can be written into coding regions without altering protein's primary structure. Moving from the start to stop codon, below we will succinctly describe the documented cases of codon context. Then, we will proceed with the description of codon context-centered tools and databases.

5.1 Cases and Reasons for Nonrandom Codon Co-occurrence

An mRNA is produced in course of transcription and interacts with various components of translation and RNA metabolism machineries such as ribosome, tRNA, spliceosome, RNA modification enzymes and various mRNA quality control mechanisms. Therefore, being a determinant of the amino acid sequence, the mRNA carries additional signals to ensure its smooth operation over the entire life cycle (Fig. 11). For example, some of the signals related to accurate and fast translation are represented by CUB, as described in the preceding section. Many aspects of mRNA biology are dependent on the codon context, and the most illustrious case is represented by translation initiation. In pro- and eukaryotes, the correct location of the start codon (usually AUG or GUG) is guided by its nucleotide context (e.g., the Shine-Dalgarno site in bacteria and archaea; the Kozak sequence in eukaryotes; see Fig. 11). Leaderless mRNAs are an exception to the rule as they have no 5'-untranslated or leader region (5'-UTR) and the start codon can be at the very 5' end. While the leader-containing mRNAs are typically recognized by a small ribosomal subunit (in complex with other translation factors), leaderless mRNA is recognized by a non-dissociated ribosome, without the involvement of protein initiation factors [147, 148]. Specific sequences downstream of the start codon are known to be required for the initiation of translation from eukaryotic mRNAs with very short or no 5'-UTR [149]. In *E. coli*, there is a notable depletion of coding sequences with codons and codon pairs that resemble the Shine-Dalgarno site. Such sequences were initially thought to be disfavored because the interaction between them and the anti-Shine-Dalgarno site of 16S rRNA induces translational pauses [150]. Recent studies attribute the aforementioned observations to a larger tendency toward the depletion of G-rich sequences, such as glycine codons [151] and suboptimal workup conditions in early ribosome profiling experiments [152, 153].

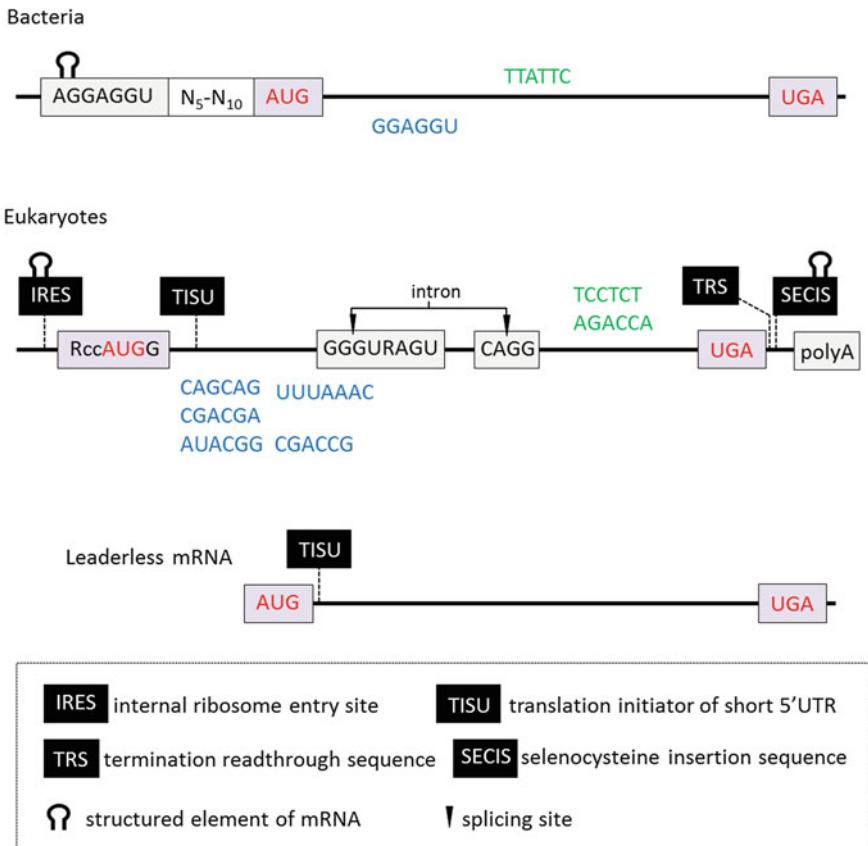


Fig. 11 Examples of mRNA sequence motifs that influence translation. The mRNAs of different origin carry different sequences that impact translation initiation, elongation and termination; see graphical legend at the bottom of the figure. Start (AUG) and stop (UGA) codons are shown in red. Shine-Dalgarno and Kozak sequences are shown in conjunction with AUG. Sequences (codon pairs) that inhibit translation or induce frameshifts are shown in blue below the mRNA. Sequences that help maintain optimal elongation rate and/or native conformation of the head of mRNA (5' region near start codon) are shown in green above the mRNA, respectively. The list of sequence motifs shown in the figure is not exhaustive and chosen to illustrate major points of the main text (see below)

Protein synthesis is rather error-prone in comparison with replication or transcription. It is estimated that amino acid misincorporation happens once for every 10^3 – 10^4 codons translated [154] and +1 frameshifts occur less than once per 3×10^4 codons [18]. Not all nucleotide sequences are equally prone to frameshifts: Proline codons CCC/U followed by C/U are particularly “slippery” in this regard. Two conditions increase +1 and +2 frameshifts at CCC-C sequence: the location of the CCC next to the start codon and the lack of TrmD-mediated methylation of guanosine residue in 37th position of the tRNA_{Pro}^{UGG} [155, 156]. The CCC-C sequences are

often encountered within the first 15 codons of a gene, coupling post-transcriptional tRNA modification to synthesis of the protein at the stage of the initiation. Not only codons and codon pairs located within the head of a gene can impact initiation: There is a marked tendency to use the codons that do not disrupt the secondary RNA structure of 5'-UTR necessary for the ribosome binding or for the protection against ribonucleases [104].

Context-dependent codon usage and its influence on translation efficiency have been first suggested on the basis of the computational analysis [157–159]. Experimental studies soon followed and provided first examples of codon pairs, such as CGA-CGA in yeast [160], and Arg or Pro codon pairs in *Salmonella* [161] that inhibited translation elongation. A principal challenge in these investigations is that modification of codon pair changes more than just codon identity or its cognate tRNA. For example, codon pair de-optimization in viral genes was initially thought to lead to attenuated viruses due to decreased rates of protein translation [162]. A more recent study explains the attenuation by perturbed replication of a viral genome, due to the artificially elevated frequency of CG and UA dinucleotides [163]. Similarly, the identity of amino acid and its context is often important irrespective of the codons being used, as it is the case for XPPX translational stalling motifs [164, 165].

Currently, the most extensive study of influence of codon pairs on translation efficiency has been reported for *S. cerevisiae* [166]. Through the analysis of the library of over 35 thousand variants of gene for green fluorescent protein randomized in positions 6–8 of the codon sequence, authors determined 17 codon pairs that inhibited translation. This inhibition was reading frame-dependent and not a result of influence on RNA secondary structure. Codons of the codon pair did not exert an inhibitory effect on translation when they were apart in the coding sequence. The order of codons in a pair is important for inhibition. Rarity and wobble decoding are the common denominators of all ten codons that make up the aforementioned 17 inhibitory pairs. The computational analysis suggests that analogous inhibitory pairs exist in human genes and could be responsible for translational pauses leading to alternative protein conformations [167]. Rare codons in the context of 3' end of a transcript in yeast were shown to slow down the ribosome and trigger Dhh1p-mediated mRNA decay [110]. Although all cited above works were focused on codon pairs, it cannot be excluded that more extensive and functionally relevant contexts exist. Indeed, synonymous substitutions within a tricodon sequence of gene *flgM* (encoding inhibitor of σ^{28} transcriptional factor needed for the production of flagellar motor in *Salmonella*) either increased or decreased FlgM activity within two orders of magnitude [168]. As in the cases described above, this study ruled out the possibility that the observed effects were due to the altered folding or stability of the synonymous versions of *flgM* mRNA. In summary, these studies represent a compelling case for the involvement of codon context in translational efficiency.

Several works provide evidence that nonrandom codon assemblies are a result of translational selection. Out of 17 inhibitory codon pairs revealed in *S. cerevisiae* genome (see above), nine are enriched in a set of orthologous sequences across five yeast species as compared to the random occurrence of constituent codons or all possible dicodons [169]. The enriched codon pairs are associated with an increased

ribosomal occupancy, implying that the primary role of such pairs is to slow down the translation at the defined loci of mRNA. The latter might be needed to ensure the proper protein folding, much like as it is documented in a number of other studies focused on individual codons [63, 170, 171] or codon clusters [172, 173]. While the identification and interpretation of non-optimal codons or codon pairs were rather straightforward, it remains unclear as to whether there is a nonrandom use of optimal codon pairs. The individual optimal codons were suggested to be located preferentially within a context of structurally sensitive sites, which would lead to protein misfolding/aggregation when mistranslated [174]. In yeast, the same subsequent amino acids in different positions along the mRNA tend to be encoded by codons recognized by the same tRNA. This so-called tRNA re-use phenomenon means that a switch to a synonymous codon read by another tRNA is less frequent than would be expected by chance [175]. Although the above case is not about the use of consecutive codons, it does make a point that highly expressed genes are under pressure to place the codons in the way that favors rapid translation.

Stop codon readthrough or frameshifting was recognized in early days of molecular biology [176, 177] and remains the best understood form of codon context [178]. The so-called post-stop sequence CARYYA was first implicated in UAG readthrough in some plant viruses [179]. It is now known that this sequence enables efficient readthrough in heterologous organisms [180]. In human cells, sequence UGA-CUAG is read through within transcript for vitamin D receptor with 6.7% efficiency [181]. Experimental approaches demonstrate the abundance of stop codon readthrough in animals, as well as diversity of codon contexts and secondary structure elements implicated in the readthrough [182, 183]. Curated collection of stop codon-recoded vertebrate genes is available at NCBI [184]. In two ciliate species, genetic codes have no dedicated stop codons, as the canonical ones are reassigned to glutamine (UAA and UAG) and tryptophan (UGA). In this species, translation termination is determined by a weakly biased usage of uridines at the flanks of the canonical stop codons and by the proximity to a polyA tail [185]. This is a striking example of a biological system where the default mode for stop codons is translation rather than termination.

Of different forms of frameshifts, -1 programmed ribosome frameshifting clearly depends on codon context. The sequence where ribosome undergoes -1 frameshift (slippery site) displays the following composition: heptamer N.NNW.WWH (N = A, G, C, U; W = A, U; H = A, C, U; dot demarcates codon borders), 1–12 nt spacer sequence and a pseudoknot structure. On the slippery site, a ribosome will be pushed 1 nucleotide back and will resume translation in -1 frame: NNN.WWW.H and so on [186]. Well-defined and rather conserved organization of the slippery site facilitated the development of a database of predicted -1 ribosomal frameshift signals in eukaryotes [187].

5.2 Anaconda—Software to Visualize Codon Pairs

The described above experimental data-driven approaches have demonstrated that codon context can influence the codon meaning, accuracy and efficiency of translation or mRNA stability. Codon pairs constitute the largest class of nonrandom contexts, which prompted the development of a specialized bioinformatics application to reveal the pairs that are represented statistically more or less than could be expected by chance. It is likely that unusual codon pairs (in terms of probability of their occurrence) are functionally loaded and are a result of selection. The application, known as Anaconda, uses annotated coding sequences as input [188]. The program is a virtual ribosome that moves from start codon in a window of three adjacent nucleotides computing the frequency of each codon c as well as codons up- and downstream of c . From contingency table of codon pair frequencies, the adjusted residuals are computed (Fig. 12). In this approach, a null model assumes that there is no codon context bias, as expected for the random codon pair distribution, taking into account background nucleotide and codon frequencies. The authors employed standard Pearson's chi-square test to probe the hypothesis of independence of codon pairs c_1 and c_2 . Once it is significant, one can see which pairs are responsible by taking the largest residuals (with absolute value >3).

All deviations from the null model captured by the application are independent of GC content and codon frequencies. A random codon pair $c_i c_j$ would be shown as a black pixel on heat map (no bias). A pair $c_i c_j$ whose frequency exceeds the one expected by chance is referred to as preferred codon pair (green pixel). Conversely, rare codon pair is the one whose frequency is below expected random value.

When applied to genome of *S. cerevisiae*, Anaconda reveals several strongly preferred and rare codon pairs [189] identified in subsequent studies as dicodons inhibiting translation, for example CUG-CUG and CUG-CGA [166, 169]. Nevertheless, the functional relevance of most of preferred or rare codon pairs remains unknown. Overall, different factors shape the pattern of dicodons in prokaryotes and eukaryotes. Some biases in bacteria and archaea can be attributed to translational selection. At least partially, the dicodon usage patterns are independent of codon usage. For example, in both domains of life there is a tendency to avoid NNU-ANN pairs as they might be a target of ribonucleolytic attack or lead to premature stop codon after frameshifting. In eukaryotes, the avoidance of GpC dinucleotide methylation in coding sequences skews dicodon usage [190].

The software offers several ways to visualize data. Codons in the heat maps can be clustered or rearranged to reflect the emergent patterns of similarity or differences. Heat maps inferred from different genomes can be overlaid to compare them or averaged to reveal major deviations from the expected codon pair usage. An example of the overlaid heat map is given in Fig. 13, produced by Anaconda on the basis of a large number of GC-rich bacterial genomes from genus *Streptomyces*. The large amount of coding sequences guarantees that each dicodon is present in the heat map (unlike in the case of a single genome; see Fig. 12) and the most pervasive and rare codon pairs are pinpointed. Here, one can clearly see that leucine codons CUC and

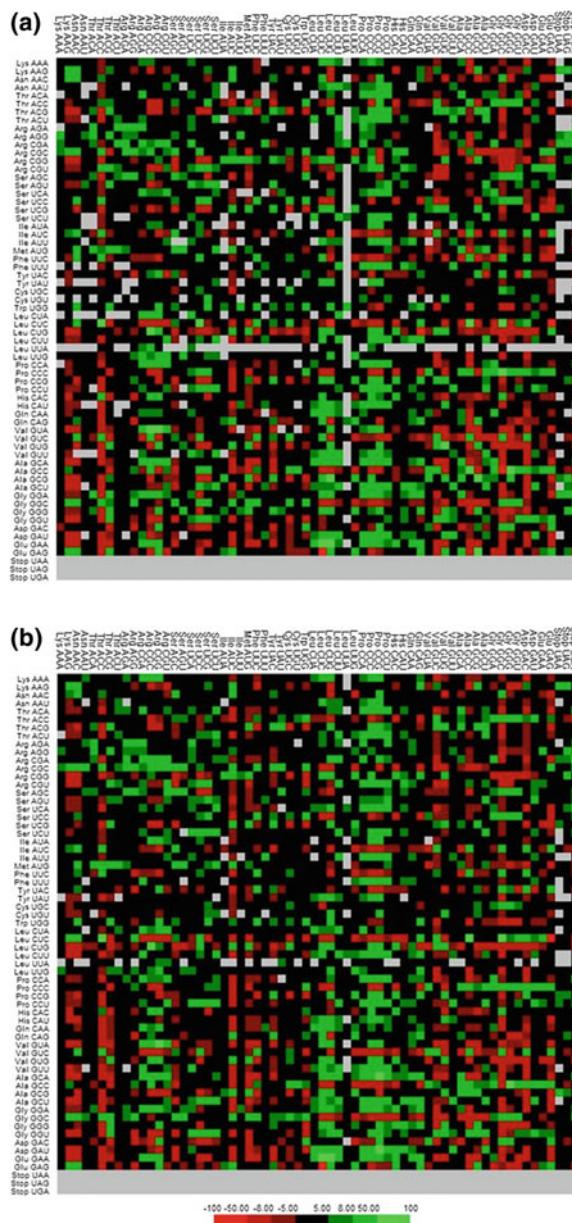


Fig. 12 A codon context heat map deduced from the dicodon analysis of *Streptomyces albus* J1074 (a) and *S. coelicolor* M145 (b) ORFomes. Logs of dicodon frequencies were the input data to build the map. Rows correspond to 5' codon (P site of the ribosome), and columns are for 3' codon. Green cells correspond to preferred contexts (occur more frequently than expected from background nucleotide frequencies) and red cells to the rare (also referred to as “rejected” or “avoided”) ones. Values that are not statistically significant are colored black (see also the main text). Gray cells indicate no data (some rare codons, such as UUA, do not occur in all possible contexts). The color scale represents the full range of values of residuals for codon context

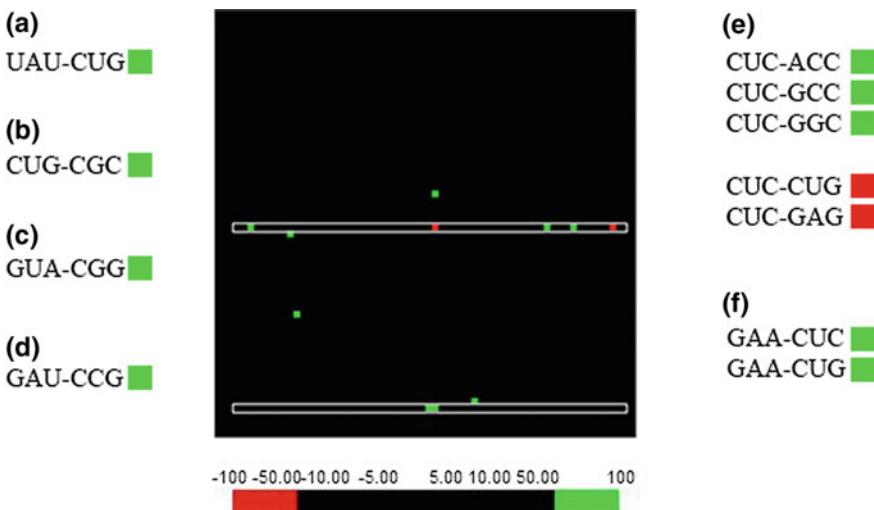


Fig. 13 Median 64×64 codon heat map of 50 *Streptomyces* genomes. In order to identify specific codon pair contexts, the map was filtered to display codon residuals that are above 50-fold difference. Green cells correspond to preferred and red cells to rare contexts. All other cases were colored black. Contexts **a–d** correspond to single pixels, and **e** and **f**—boxed lines of the map. See Fig. 12 for more details

CUG are overrepresented among the codon pairs conserved across 50 genomes. Out of 11 such pairs, nine carry either of the two aforementioned codons. The pairs can be summarized as follows: NNU-CNG and GNA-CNN. There are also two patterns of dependency between non-adjacent bases: CUS-NSN and CUS-NWN. Some of the observed patterns have already been observed in other taxa belonging to all three domains of life. These cases include avoidance of CUC-CNN or preference for NNC-ANN [191]. Specific biological reasons for the observed patterns await their verification.

5.3 Codon Utilization Tool (CUT)—A Database of k-Codon Usage for Yeast, Rat and Mice

Statistically rigorous description of different kinds of unusual codon patterns is a valuable starting point to investigate their biological significance. This kind of information is especially important for model eukaryotic organisms such as yeast *S. cerevisiae*, mouse *Mus musculus* and rat *Rattus norvegicus* that possess complex genomes and are subjected to global research. CUT database (<http://pare.sunycse.com/cut/index.jsp>) has been developed to compute and visualize codon combinations, from di- to pentacodons, in the aforementioned species [192]. The resulting datasets can be filtered according to user-defined Z-score level to focus on the most biased (enriched)

or depleted as compared to expected frequency) k -codons. In addition, user may download any sequence of interest to enumerate the constituent k -codons. The CUT database was used to reveal transcripts dependent on conditionally modified tRNAs or those with internal stop codons UGA that might be subjected to programmed readthrough. There is increased realization that it is important to take into account codon pairs in various biological processes. Consequently, a comprehensive online resource for codon usage HIVE [126] was recently expanded with Codon and Codon-Pair Usage Tables (CoCoPUTs) database [193]. The latter lists frequencies of codon pairs and dinucleotides for all completely sequenced genomes in GenBank.

6 Synonymous Codon Usage Bias (CUB)

While the formal definition of CUB was given in Sect. 4, here we focus on the intrinsic CUB, caused by the general benefit of a certain codon over its synonyms in terms of efficiency and/or accuracy of translation, or other properties (such as, but not limited to, production or stability of mRNA). In contrast, the context-dependent CUB arises when the effects of a codon depend on the neighboring sequence, in either a short or a long range. This kind of codon usage bias (unrelated to different decoding or other functional properties of synonyms) was discussed in Sect. 5, and it will not be considered here. Then, we will proceed with the description of the relevant visualization tools.

6.1 CUB Types and Plausible Biological Reasons

The CUB is detectable at different levels of organization of genetic material. In different species, usage of synonymous codons correlates with GC content. For example, in *Lactobacillus acidophilus* (GC content $\approx 50\%$) two lysine codons AAG and AAA are encountered at the same frequency, while in *Streptomyces coelicolor* (72% GC base pairs) AAG heavily dominates (98%). Differences in CUB between the species are believed to be caused mainly by mechanistic neutral processes such as mutations [67, 194, 195] and/or biased gene conversion [196, 197]. Nevertheless, variation in GC content is unlikely to be caused solely by different mutational patterns and recombination in bacteria, implying the contribution of selection [198, 199]. A combination of neutral and selective factors is also thought to shape synonymous codon usage in eukaryotic genomes [200].

Different genes within a given genome can be enriched in different synonymous codons. This intergenic CUB is thought to be a result of selective forces, of which translational selection is perhaps the most extensively studied. The most compelling case for translational selection comes from the analysis of highly expressed genes of bacteria having large effective population sizes. Under such conditions, translational selection will favor the use of synonymous codons for which there is high

concentration of cognate tRNAs in the cell. The codons selected for by the described above adaptive process are referred to as optimal. In the scientific literature, such codons can also be termed as frequent, fast or popular, putting special emphasis on one facet of codon property. In line with the above, infrequent (rare) codons are often referred to as slow or non-optimal ones. It has to be noted that meaning of codon optimality is an evolving concept as researchers develop novel approaches to measure gene expression and tRNA abundance [64, 86, 106]; different sets of codons can be viewed as optimal under different conditions [26, 86]. In a classical view, optimal codon improves all or a combination of the three following parameters of protein synthesis: speed of translation, its accuracy and robustness to mistranslation. The first two aspects have already been mentioned in Sects. 4 and 5; the latter postulates that mistranslation is very costly to the cell as it may lead to toxic misfolded proteins. Consequently, a coding sequence is under constraints to use those codons which would help tolerate or minimize missense mutations [201, 202]. Current advances in high-throughput system biology approaches largely substantiate the idea that optimal codons improve heterologous protein production [101]. As optimal codons are translated faster [203–205], this would increase the translational efficiency, e.g., number of protein molecules produced per transcript. Nevertheless, the effects of codon optimality on translational efficiency of endogenous mRNAs remain less clear-cut [89, 206]. A number of other observations point to the fact that our understanding of causes and effects of CUB is oversimplified. No significant differences were revealed in ribosome occupancy of either optimal or non-optimal codons in several model systems [64, 86, 150], pointing to the possibility that in native genes both classes of the codons are decoded at a similar rate. Focus on the abundance of cognate tRNA in determining translation accuracy of a codon might also be misleading, as the large pool of near-cognate tRNAs could outcompete cognate tRNA [207]. As mentioned above, much of our knowledge of translational regulation comes from the bacterial systems. In animals, strength of translational selection is weaker, yet there is evidence for non-neutrality of synonymous mutations [208] as well as for their involvement in translation modulation [209, 210].

Adaptive processes other than translational selection also contribute to the CUB. Genes might be depleted of certain codons to minimize the formation of mRNA secondary structures near the 5' end, facilitating the translation initiation [83, 104]. A different speed of ribosome movement along mRNA as a function of codon optimality evokes additional forms of selection. It is known that optimal codons increase the stability of mRNA in bacteria and eukaryotes [132], whereas non-optimal ones serve as a signal of mRNA decay. Here, codon optimality therefore refers not only to translational efficiency of certain codon, but also to its capacity to encode mRNA stability. Rare codons are placed in protein domain linker regions, whereas optimal codons are grouped within functional domains. Most likely, slow ribosomes on linker regions enable proper co-translational protein folding [63, 86, 171]. In fungi, codon usage was shown to impact transcription through the mechanisms independent of mRNA translation or stability. Particularly, in *Neurospora* the CUB correlates with mRNA and protein abundance, and codon optimization in this species increases mRNA and protein levels. How codon identity influences transcription is unclear. The authors

suggest that DNA sequences overlapping mRNA coding regions contain elements recognized by the transcriptional machinery, and synonymous replacements alter these sequences, thus impacting transcription [211]. Finally, other as-yet-unknown translation-independent factors could dictate codon choice, as recent work on toxicity of certain synonymous versions of mRNA has shown [212].

Distinct CUB patterns are revealed within coding sequences. A head of the gene is enriched in rare codons, presumably forming a ramp for slow loading of ribosomes onto mRNA. The latter helps avoid ribosome collisions and improve the translation initiation [120, 213]. As the initiation is rate-limiting step of translation [128], there was a debate as to why the codon content is so evidently nonrandom along the entire transcript. For example, last 50 codons of eukaryotic mRNAs are often highly optimized [120]. One possibility is that ribosome pausing on sites distal to 5' end of the gene can somehow be relayed back to the head of the gene, thus preventing new round of initiation at the start codon [214]. Under this scenario, the initiation and elongation steps are linked, and transcript-wide enrichment in optimal codons has an adaptive value. Not taking into account this factor might lead to alternative conclusions that only initiation is the major determinant of mRNA stability [215].

In summary, experimental data reveal that synonymous substitutions in coding regions may affect all steps of gene expression, such as replication, transcription, translation, mRNA stability, toxicity and protein folding. Moreover, codon optimality is proposed to be predictive of protein half-life [216]. It is impossible to develop an experimental model where the synonymous codon replacement would be related to a single aspect of gene expression or mRNA structure; we must deal with inherent complexity of biological systems. These challenges notwithstanding, there is undoubtedly progress in our understanding of relationship between codon optimality and translational efficiency, at least in bacteria and model eukaryotic systems such as yeast and human cell lines. An overall conclusion is that mRNAs of highly expressed genes are composed of optimal codons that enable accurate and fast translation; the latter is also important for mRNA stability. Codon optimality appears to be relative to some extent, as different environmental conditions modulate the availability of charged tRNA for a given codon [28]. Non-optimal codons are important, too. They can be used to decrease the mRNA stability where needed and to either improve [203] or derail [217] the co-translational folding. The influences that codons exert on translation will depend on their context (see preceding section) as well as qualitative and quantitative parameters of tRNA pool [28, 166, 218, 219].

6.2 Visualizing CUB Between and Within Genes

The data on genome-wide codon usage can be inferred from annotated genome sequence or found in databases such as Kazusa or HIVE-CUT [126], and converted in various types of bar charts or diagrams. For example, CUB for different genomes can be conveniently visualized in the form of circular diagrams. The latter grasp the

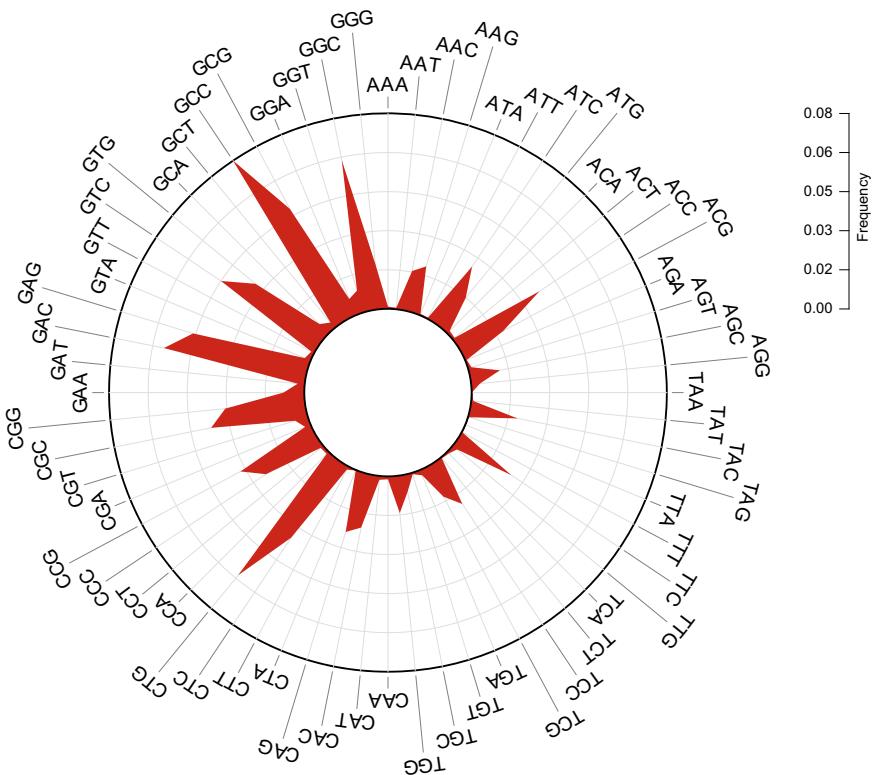


Fig. 14 A circular diagram of codon usage in *Streptomyces coelicolor* A3(2) genome (GC content 72%). Codon counts within each gene were summed over the entire genome and converted into the diagram. The codon usage is skewed toward GC-rich triplets

most salient features of codon usage for a given genome, such as preferred usage of GC-rich codons in GC-rich genomes (Fig. 14).

Codon indices can be employed to visualize the differences in CUB between two or more genes. In this regard, CAI remains the most popular index among reference-based measures, while Nc or codon bias index (CBI) [1] is often used if one prefers reference-free approach (see Sect. 4 for more details). Care should be taken when interpreting the differences in codon indices for individual genes, as they may be caused by factors unrelated to translation efficiency (see above). Ideally, measures of CUB should be applied to genes whose evolutionary history, expression levels and function are well understood.

Positional CUB has been reported in many studies, and it remains a fertile ground for the development of dedicated visualization tools. For example, extreme depletion of leucyl TTA codon in GC-rich *Streptomyces* genomes is accompanied by positional skew of this codon toward 5' end of the gene [220]. Based on this finding, a Web service TTALynx was developed that visualizes positional codon usage bias on a

genome-wide scale [221] as depicted in Fig. 15. The statistical significance to the observed biases remains unaddressed. Perhaps, one way to bypass this problem is to aggregate the data from multiple genomes to see the global trends. Expected distribution can be inferred from simulated coding sequences that preserve nucleotide and codon biases of the genomes under study (Fig. 16).

7 Codon Substitution Models (CSMs)

Codon context and codon usage biases can be viewed as a horizontal “axis” of codon changes, those that happen in a single coding sequence. In contrast, the vertical “axis” is represented by changes that take place over time in different homologous genes. Namely, once two copies of the gene occur due to either duplication or speciation event, they start diverging due to the accumulation of various random genetic changes, such as insertions, deletions, translocations and recombination. However, the most frequent type of rearrangement is point mutation, which is a change of a single DNA base to another one. Depending on the experimental approach, the point mutation rates in *E. coli* were reported to be within the range of $(0.25\text{--}5.0) \times 10^{-10}$ per generation [222]. If a mutation spreads in a population, because of either a genetic drift or an adaptive advantage it confers to organisms, becomes fixed and is called a substitution. For a set of aligned DNA sequences coding for protein, the substitutions can be modeled at three levels: nucleotide, amino acid and codon ones (Fig. 17). It makes sense to use codon substitution models (CSMs) in order to have the alignments to account for the structure of the genetic code, unequal biases at three codon positions and selection on protein [223]. All of the aforementioned aspects can be modeled explicitly by CSMs. The latter are recent addition to the toolkit of computational biology as compared to nucleotide and amino acid models, which is partly explained by the absence of relevant and sizable datasets in early days of molecular biology. The availability of homologous genome sequences from closely related species stimulates the interest in CSMs as they offer detailed view of the forces that shape protein-coding sequences. In this section, we begin with succinct description of the founding principles that are at the core of all CSMs. Then, we describe several most popular CSMs as well as recent developments not covered in other reviews. An overview of tools for simulation of coding sequences and visualization of CSMs will conclude this section. For a deeper acquaintance with the topic of CSMs, interested reader is referred to [1, 224].

7.1 Basic Concept

Typical substitution models of biological sequences, including CSMs, are defined as Markovian processes. So, a stochastic evolutionary process over continuous time is defined on (codon) states, where the next state depends only on the current state

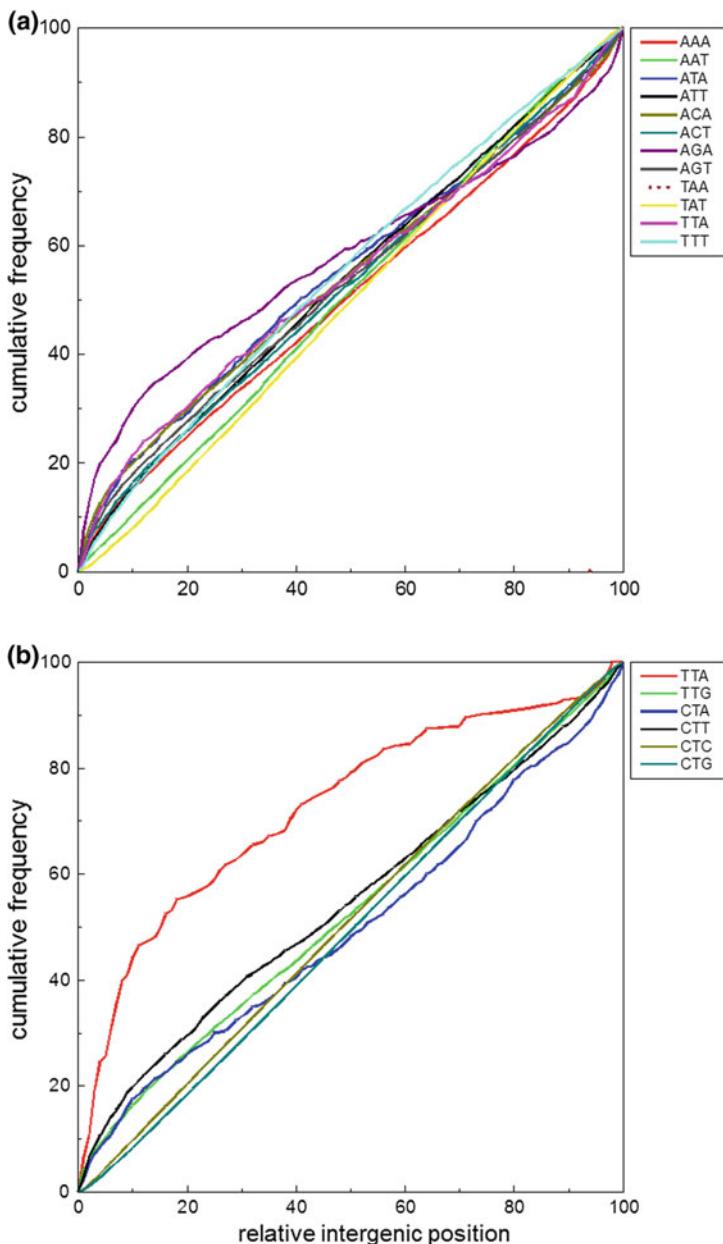


Fig. 15 Cumulative distribution of selected codons within chromosomes of *Rhodopseudomonas palustris* (a; 65% GC) and *Streptomyces griseus* (b; 72% GC). Both X- and Y-axes are given in %. Straight line means that no skew is observed and codon distribution is uniform across coding sequences. Upward shift is an indicative of preferred usage of codons closer to start codon; downward shift marks preferred clustering of codons closer to stop codon. Positional bias of TTA codon usage within *S. griseus* genome is clearly observed: More than 50% of all TTA codons (around 80) in this species are found within first 20% of length of its genes

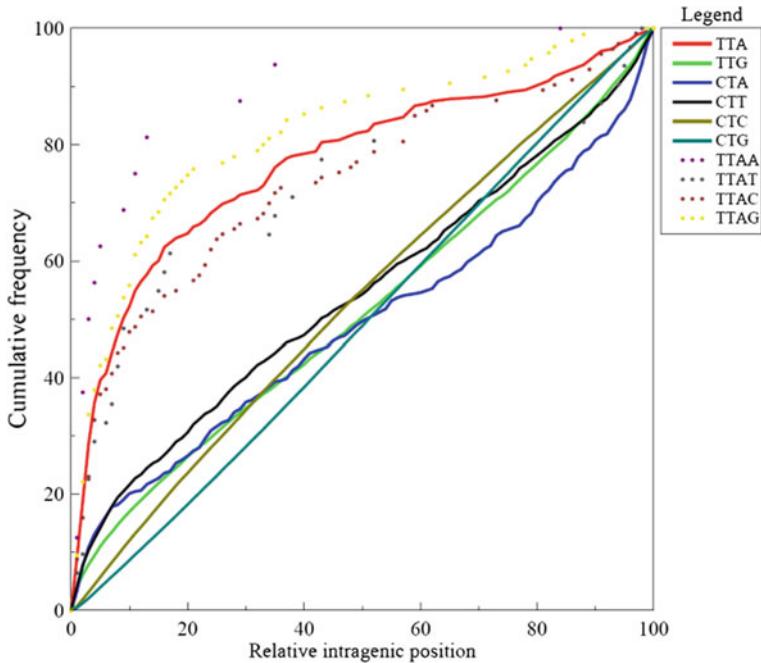


Fig. 16 Cumulative distribution of leucine codons and leucine codon-based quadruplets (TTAN) within genome of *Actinosynnema mirum* (73% GC). See Fig. 15 for more details

and not on any past states (i.e., Markovian property). Assuming that seq1 in Fig. 17c represents a parental sequence, the state of first codon CCT in seq2 depends on the present state of the first codon (CCG) in seq1. In Markov models, each character in a sequence (nucleotide or amino acid residue, or codon) is a random variable that undergoes substitutions independently and with the same probabilities as every other character. In a coding sequence, each codon is described by a single Markov chain over 61 codon states (sense codons). A Markov model is defined by the generator matrix $Q = \{q_{ij}\}$ of instantaneous rates of change between the sense codons i and j , i.e., describing the pattern of change when time has no past. Substitutions involving stop codons are prohibited. To compute the probability of various transitions between codon states over time t , a probability transition matrix $P(t)$ can be computed by solving the differential equation:

$$\frac{dP(t)}{dt} = Q P(t) \quad (31)$$

At timepoint $t = 0$, the probability of a change is 0 (off-diagonal values) while the probability of no change is 1 (values on a diagonal). Therefore, $P(0)$ is an identity matrix I . This provides Eq. 31 with an initial condition, yielding the following solution:

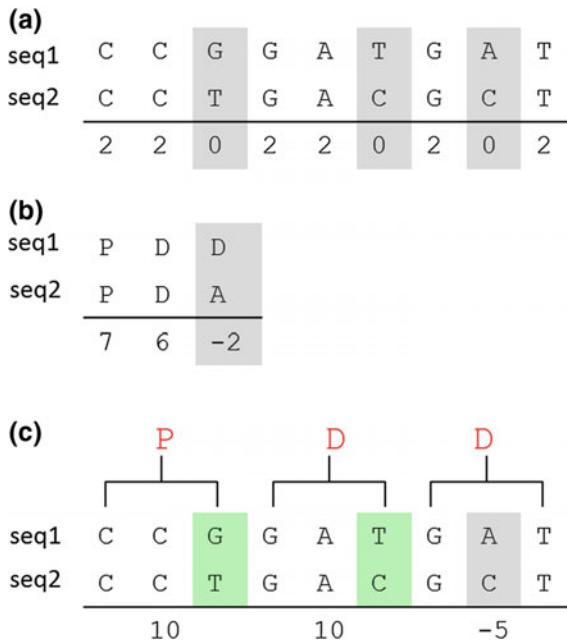


Fig. 17 The substitutions in an aligned pair of homologous protein-coding sequences can be assessed at the nucleotide (a), amino acid (b) and codon (c) levels. In case of nucleotide alignment, all matches will have arbitrary score 2, and all mismatches (gray background) would be zero. For amino acid alignment, the scores were taken from BLOSUM62 model. In codon alignment, synonymous substitutions are highlighted with green background; scores are from [225]. Square brackets mark codons and amino acids (red) they encode

$$P(t) = e^{Qt} \quad (32)$$

Therefore, given a rate matrix Q , one can compute $P(t)$ for any time $t \geq 0$, which contains probabilities $p_{ij}(t)$ of change from state i to state j over time t .

In order for Q to define the probability matrix $P(t)$, it should satisfy the following mathematical properties. All its off-diagonal entries have to be nonnegative, and sum of each row should be zero. Therefore, the diagonal entries $q_{ii} = -\sum_{i \neq j} q_{ij}$ are negative. Q is also assumed to be independent of t (e.g., is time homogeneous). This means that the same rate matrix will be globally used over all branches of tree representing given multiple sequence alignments. Most of CSMs are formulated as time reversible—that is, direction of change between i and j cannot be told: $\pi_i p_{ij}(t) = \pi_j p_{ji}(t)$. It is not biological reality but rather mathematical simplicity that drives the widespread assumption of time homogeneity and reversibility. For example, a full empirical irreversible model for N states (e.g., codons) has $N \times (N - 1) - 1$ free parameters, whereas for a reversible one model this number is reduced to $N \times (N + 1)/2 - 2$. In addition, an reversible and homogeneous model does not require the knowledge of the root placement, which is often non-trivial.

CSMs can be parameterized in different ways (e.g., with substitution rates between synonymous and non-synonymous codons, with transition/transversion rates, etc.), and these parameters can be inferred from observed codon sequences. The CSMs can be used to compute the probability that, over the evolutionary time spanning the input multiple sequence alignments, a coding sequence will accumulate substitutions resulting in the observed sequences, given the model. This is essentially the likelihood L of the observed data D , which is proportional to the probability of D given the substitution model M with its parameters: $L = p(D|M)$. Model parameters and evolutionary divergencies (i.e., phylogenetic branch lengths) can be estimated by maximum likelihood or using the Bayesian approach [224, 226, 227, 228].

New models and methods have to be compared against the existing ones to understand potential caveats and advantages of the former. Likelihood-based tests are a relatively simple and popular way to compare models [229]. Models that provide a better description of the unknown process which produces the data typically have a higher likelihood when adjusted for a number of parameters they contain. It is important to make sure that the higher likelihood arises due to better explanatory power of the model and not due to overfitting. It is customary to compare a model with more parameters M_A to a simpler null model M_0 . If a complex model can be transformed into a simpler one by constraining some of its parameters, then the models are nested and a likelihood ratio test (LRT) can be applied:

$$\text{LRT} = -2 \log\left(\frac{L(M_A)}{L(M_0)}\right) = -2(\log(L(M_A)) - \log(L(M_0))) \quad (33)$$

The p -value can be computed for a LRT statistic to assess the significance of the difference between the optimized log-likelihoods $\log L(M_A)$ and $\log L(M_0)$.

If the models are not nested, then the Akaike or Bayesian information criteria (AIC or BIC, respectively) can be used to rank the models—but not to reject one model in favor of the other. For example, AIC for model M_A would be:

$$\text{AIC} = 2k - 2\log(L(M_A)) \quad (34)$$

where k is the number of free parameters [230].

Simulations are another way of analysis and comparison of CSMs. In this approach, an initial set of random sequences is generated according to the specified parameters. Then, evolution of the set is simulated using the substitution models of interest. The evolved set of sequences is used to learn about the model and parameters that produced the former. Any inferences about the past of the evolved sequences can be compared to the true parameters used in course of simulation. The results of simulations can be visualized in the form of receiver operating characteristic (ROC) curves. The latter allow for comparison of various properties of the models, such as specificity or sensitivity, on different datasets. The perfect predictors would display high sensitivity with low false-positive rate, and the curves will be attracted to the upper-left corner (Fig. 18). One has to be careful, however, in interpreting the results of simulations, as they often oversimplify the real evolutionary process.

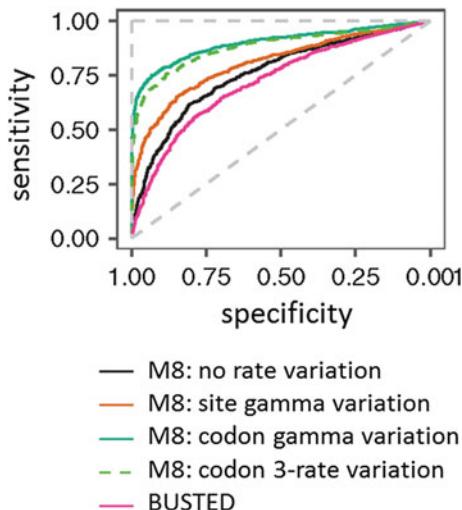


Fig. 18 ROC of four M8-based and busted CSMs on a dataset with codon gamma rate variation. Specificity is defined as the proportion of correctly identified alignments simulated under a model with positive selection, and sensitivity is defined as the proportion of correctly identified alignments simulated without positive selection. The dashed diagonal line shows the theoretical performance of the random predictor, and the dashed vertical and horizontal lines indicate the theoretical performance of the perfect predictor. This figure is part C of Fig. 1 from Davydov et al. [231]

It is important to test new methods on real datasets. A CSM can be applied to a collection of orthologous genes from different species to infer the time of speciation. Different sets of orthologous genes are expected, ideally, to yield the same time of separation. In practice, there will be a distribution of speciation times inferred from different datasets. Datasets themselves and methods of distance measurement contribute to this variance. Hence, when different methods are applied to a dataset, then variance in results indicates differences in method performance. Optimal methods would provide an estimate with the lowest variance (but low variance does not mean that the method is optimal as it may be also a sign of a systematic bias). If phylogenetic reconstruction is the primary area of application of a model, then it can be challenged with the sets of sequences for which true phylogeny is known. The best method would be the one that recovers the highest number of correct trees from the reference datasets.

7.2 Diversity of CSMs

Two principally different types of CSMs can be distinguished. The first type is referred to as mechanistic or parametric one. These models describe all possible

types of codon substitutions with a defined and finite set of parameters. In contrast, empirical CSMs are derived from large sets of aligned homologous sequences. It has to be noted that parameters of mechanistic models themselves are based on empirical data. For example, codon frequencies can be inferred from nucleotide frequencies. Hybrid, semiempirical CSMs are quite widespread: These are built as empirical and then amended with certain parameters. Parametric models are especially useful in dissecting the factors and mechanisms of coding sequence evolution. Empirical models are suitable in the area of phylogenetic reconstruction. Exhaustive description of all available CSMs and their computer implementation is beyond the scope of this section as this topic is deeply covered in the literature [1, 224, 232]. Here, we want to get the reader acquainted with the principles of development of CSMs and the ongoing efforts to more fully incorporate the biological reality into the models. Historically, parametric models were developed first and thus will be considered first; then, we proceed to empirical and semiempirical models.

One-ratio model M_0 is the simplest parametric CSM, where the instantaneous rates $Q = \{q_{ij}\}$ are provided as follows:

$$q_{ij} = \begin{cases} 0, & \text{if } i \text{ and } j \text{ differ by more than 1 nucleotide substitution} \\ \mu\kappa\pi_j, & \text{if } i \text{ and } j \text{ differ by 1 substitution; synonymous transition} \\ \mu\kappa\omega\pi_j, & \text{if } i \text{ and } j \text{ differ by 1 substitution; nonsynonymous transition} \\ \mu\pi_j, & \text{if } i \text{ and } j \text{ differ by 1 substitution; synonymous transversion} \\ \mu\omega\pi_j, & \text{if } i \text{ and } j \text{ differ by 1 substitution; nonsynonymous transversion} \end{cases} \quad (35)$$

where μ is the scaling constant to make sure that the average substitution rate: $-\sum_{i=1}^{61} \pi_i q_{ii} = 1$; π_j is the equilibrium frequency of codon j calculated from observed nucleotide frequencies at the three codon positions; κ is the transition/transversion rate ratio ($\kappa = 1$ means no transition bias); ω is non-synonymous/synonymous substitution rate ratio (d_N/d_S) describing selection on non-synonymous mutations. In M_0 , double and triple substitutions within a codon are prohibited, and π , ω and κ are estimated from data to characterize processes at the DNA level. The example of M_0 above represents the so-called GY-type CSM assuming that q_{ij} rates are proportional to the frequency of a target codon π_j . In contrast, in MG-type CSMs q_{ij} is proportional to the frequency of the target nucleotide f_x^P [1, 224]. A detailed treatment of the ways to compute codon frequencies under different models is given in [233].

Detecting the selection on a coding sequence was a primary goal of the CSM development; this point will be further elaborated in the next section. The M_0 model assumes constant ω across sites in the input alignment, which is not biologically justified. Selection, positive or negative, often impacts different lineages and codon sites in a sequence in a different way, since the selective pressure is different due to protein folding constraints and the changing evolutionary pressures over time. Consequently, new models were proposed to account for variation in selection. Using model tests mentioned above, one can deduce whether the models allowing variability

in ω explain the data better than the restrictive one ω -ratio model, and if so, what types of selection may operate on the sequences. The branch models allow variation over the time, assigning a different ω for different branches of phylogenetic tree. Site models provide variation of ω between codons through various distributions of ω or its components, d_N and d_S .

Through adding new parameters and using different parameterizations, increasingly sophisticated CSMs can be constructed. For example, rates of double and triple substitutions can be set to some nonzero value [234]; different rates for different codon pairs can be proposed; different positions of a codon may have different rates as well. We list some example below.

Besides selection at the protein level, a wide specter of CSMs is now available that are tailored to assess other properties of coding sequences. Models have been developed to detect selection for amino acid properties [235]. Here, amino acid substitutions are divided into conservative (change leads to an amino acid similar in its physicochemical properties) and radical (switch of the properties). Genetic algorithm, known as **CodonTest**, has been developed to estimate such models [236]. A number of models aim to reveal site interdependency (both within [237] and between [238] codons) and to study codon bias [239].

One-ratio CSM can be combined with a model of selection on synonymous substitutions that takes into account codon-specific nonsense error rate [240]. An interesting aspect of this nested model is that it gives clues about protein expression levels. The latter parameter of gene expression can also be estimated using **SelAC** [241]. SelAC is an CSM built using the cost-benefit function that links gene expression level to the strength of stabilizing selection. This is CSM where gene and amino acid-specific substitution matrices are nested within a substitution model. The genome-wide parameters used by the model are: nucleotide-specific mutation rates scaled by the effective population size; amino acid side chain physicochemical weights; and a gamma distribution shape parameter describing the distribution of selective strength. There is also a gene-specific parameter ψ describing the average rate at which the protein is produced. SelAC can be seen as the latest attempt to marry codon and population genetic models; both deal with stochastic process of character substitution, although the former focus on interspecies variations, whereas the latter explores intraspecies variation [242]. **SENCA** is another interesting example, defined as a CSM that separately considers mutational biases, CUB and amino acid preferences [243]. It represents a growing roster of mutation-selection models that attempt to account for selective patterns at the amino acid level, with some approaches allowing for heterogeneity in these patterns across codon sites. Basically, change in a nucleotide sequence is viewed as the product of the probability of a mutation from codon to another, the probability that the mutation becomes fixed in the population (which, in turn, depends on population parameters, most notably effective population size) and a scaling constant [244, 245]. Mutation-selection models pave the way to study selection on coding sequences, codon usage bias and codon/amino acid interdependence. SENCA confirmed several previous observations, such as the universal mutational bias toward AT bases in bacterial genomes. Finally, a general-purpose parametric (**GPP**) model has been proposed that includes double and triple

mutations and multiple non-synonymous mutation rates [246]. Including multiple mutations per codon was shown to increase accuracy and power of CSMs. Existing parametric codon models are considered as special cases of GPP.

Empirical CSMs were developed once datasets of homologous genes became large enough to reliably calculate all possible codon transitions. First such matrix was reported in 2005 using 8.3 million aligned codons of vertebrate DNA [225]. In building this CSM, the authors closely followed the methodology used by M. Dayhoff to construct the seminal amino acid substitution PAM matrices [247, 248]. Rapid accumulation of genomic sequences stimulated the development of other empirical CSMs as well as improved methods of their assessment [249]. Principal component analysis of 3666 empirical CSMs estimated from multiple alignments of mammalian genes revealed that ω and multinucleotide codon substitutions are two the most important factors for codon models [250]. At the same time, transition/transversion ratio was not found to be an important component. The authors also suggested that synonymous substitutions within the serine box requiring two changes might be better modeled as non-synonymous. Mixed mechanistic-empirical (or semi-parametric) CSMs soon followed. For example, an empirical amino acid substitution matrix can be used to expand to a semi-parametric codon substitution model [251, 252]. Such combined models were fitted to real datasets and shown to be superior to purely empirical models. Similar results were obtained upon the examination of semi-parametric CSMs enhanced with parameters inferred from above-mentioned principal component analysis [253]. The area of immediate use of such semi-parametric CSMs is the scoring of codon alignments of genes from taxonomic groups used to train the model.

7.3 *Simulation of Codon Substitution Patterns*

In silico generation of random coding sequences according to a certain model (i.e., simulation) is an indispensable tool for molecular phylogenetics, evolution and population genetics. First, simulated sequences are needed to test the behavior of CSMs and to estimate evolutionary parameters (vide supra). These could help understand evolutionary or genetic consequences of the processes that are not analytically tractable. As a result, a number of genetic sequence simulators have been developed; those available before 2012 are reviewed in [1, 254, 255]. Below, we briefly describe a conceptual framework of coding sequence simulation and then consider novel developments in this area.

The evolution of either real or artificial sequences coming from one (population data) or different (phylogenetic data) species can be simulated forward and backward in time. The forward-in-time simulators generate ancestral sequences and describe their evolution from the past to the present. **GenomePOP** and **SFS_CODE** are examples of such simulators of coding sequences. Backward-in-time simulators describe genealogical process that connects genes sampled from a population to their most recent common ancestor. These simulations are also known as coalescent because the

lineages coalesce progressively as a function of modeling parameters. **CodonRec-Sim** and **NetRecodon** exemplify coalescent simulators; the latter allows intracodon recombination. For generating population data, coalescent simulators are more efficient because they monitor the evolutionary trajectories of sample genes, not all members of the population. However, backward approaches are limited for studies of the effects of natural selection in populations. Forward approaches can be used to model complex evolutionary scenarios and estimate various parameters, such as ω , in the presence of confounding factors, e.g., recombination [256]. In all simulation approaches, codon substitutions can be introduced at random independently among different sites.

Several codon-aware simulation programs became available in or after 2012. **ALF** is a user-friendly Web-based application (<http://alfsim.org>) aimed to simulate sets of genes under different models (e.g., with different rates); genetic rearrangements can be simulated at both genome and gene levels [257]. For coding sequences, one can model the substitutions at nucleotide, codon and amino acid levels under specified conditions (GC content, gene duplication, loss or fusion, etc.). **SWG**E software package simulates evolution of sequences genome-wide, allowing for different substitution models along the sequence, as specified by the user [258]. It is possible to recapitulate complex coalescent life histories that include recombination, demographic changes, migration, etc. **SimPhy** is a software package for phylogenomic simulation at species, locus and gene levels [259]. **SimPhy** can be used to simulate codon alignments under different initial assumptions. **SLiM 3** is an evolutionary simulation framework for forward population genetic modeling [260]. Developers of SLiM 3 invested much effort in making their program package user-friendly and flexible. First, this is a scriptable framework that neither limits the user to a fixed set of simulation options nor requires fluency in a specific programming language (as most earlier programs do). Authors have chosen Eidos language (similar to *R*), which operates high-level concepts such as “for” and “if” and is supposed to provide an easy way to coding custom simulation scenarios. Further, SLiM 3 provides an interactive scripting interface for visual debugging of the simulation models as they run in real time.

7.4 Approaches Toward Visual Representation of CSMs

There is not much room for visualization of matrices describing parametric codon models in contrast to the empirical ones, where tabular data are a typical raw output [225]. The patterns of character substitution are usually visualized using bubble plots where both bubble size and color inform about the frequency and nature of substitution, respectively. This kind of visualization has been applied to visualize amino acid [233] and codon [250] substitutions. Particularly, codon substitution rate parameters can be estimated from multiple codon sequence alignments with expectation-maximization procedure [228] implemented in program **XRate** [261, 262]. Codon alignments can be prepared in several ways. For example, if the reading frame is

known, one can start with coding DNA sequences, translate them to amino acid sequences, align these and then back-translate into codon alignments [263]. Today, however, more sophisticated tools are available; **MACSE** builds coding alignments that account frameshifts and stop codons [264].

Figure 19 presents a typical visualization of an empirical CSM inferred from a set of 40 orthologous genes for sporulation-specific protein SsgA in *Streptomyces* [265]. Here, one can observe a unique pattern of substitution, which will be distinct for each dataset, as Fig. 20 demonstrates. Noteworthy, the latter shows that substitution pattern is different for orthologous groups of transcriptional factor AdpA [266] from different actinobacterial genera. Although the above figures focus on orthologs, it is possible to visualize the substitution model for reasonably aligned sets of sequences of any origin (such as paralogs).

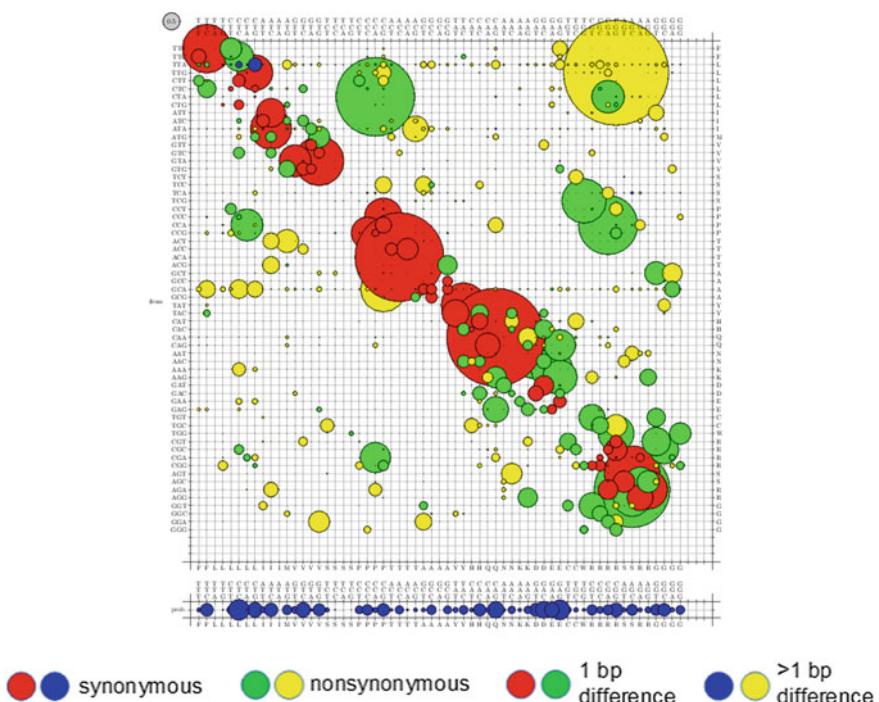


Fig. 19 Bubble plot of codon relative substitution rates computed with the help of XRate for orthologous group highly conserved in *Streptomyces* sporulation-specific genes *ssgA* (145 codons; accession number for *S. venezuelae* *ssgA* is ACC77837). Codons are ordered according to biochemical similarity of respective amino acids. The size of a bubble reflects the rate of substitution of one codon (rows) by another (columns). Synonymous one-nucleotide and two-nucleotide substitutions are shown in red and blue, respectively. Non-synonymous one-nucleotide and multinucleotide substitutions are shown in green and yellow, respectively. Blue diagram (below the matrix) shows overall codon usage. All plots in this and the following figures are equally scaled (gray circle, top left corner of the plot)

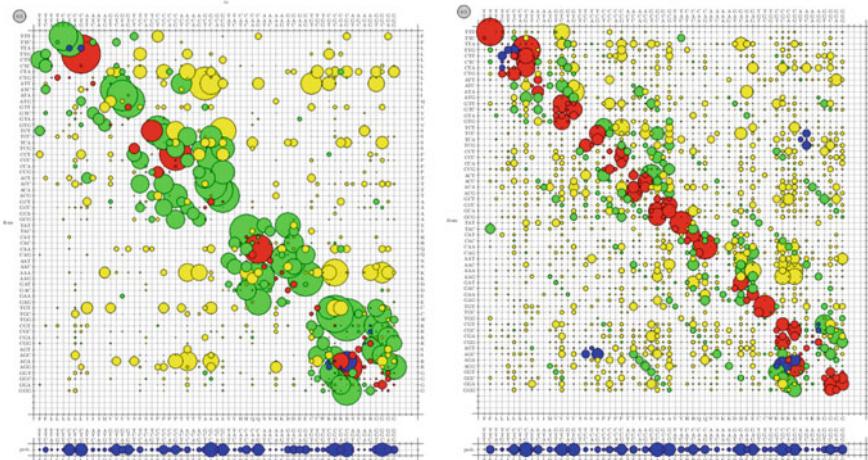


Fig. 20 Bubble plot of codon substitution rates computed with the help of XRate for orthologous groups of genes for transcriptional factor AdpA from *Streptomyces* (left) and *Streptosporangium* (right). For each model, 30 codon sequences were used. The size of a bubble reflects the rate of substitution of one codon (rows) by another (columns). See Fig. 19 for color code

Cursory analysis of bubble plots reveals salient features of evolution of the underlying sequences. Clustering of substitutions along the diagonal line reflects the absence of substantial amino acid changes, because codons in the plot are ordered according to their physicochemical properties. The plots depicted in Figs. 19 and 20 show that the two datasets experience substitutions to a different degree. This may imply that respective coding sequences are under different selective pressures to maintain their primary structure. Symmetry is another important feature of the bubble plots. The latter is not apparent in the first two figures; however, pronounced symmetry is clearly seen in bubble plot generated on the basis of RuBisCo proteins from flowering plants (Fig. 21). This suggests unusual tolerance of RuBisCo to certain kinds of substitutions; e.g., both forward and reverse changes happen at the same frequency. Such a feature of essential genes may render them more robust in the face of mutations and mistranslation [201]. Low incidence of near-diagonal substitutions for RuBisCo genes is another distinctive feature in Fig. 21, probably reflecting the action of stabilizing selection on RuBisCo coding sequences. Physicochemical properties of amino acids to a large extent dictate the substitution patterns. Nevertheless, some of the patterns remain unexplained. For example, it is puzzling that RuBisCo genes (*rbcL*) are depleted of synonymous substitutions. Given the enormous importance of RuBisCo for the biosphere [267], it will be interesting to study larger and taxonomically distinct *rbcL* datasets by comparing their patterns of codon substitutions based on the bubble plot analysis.

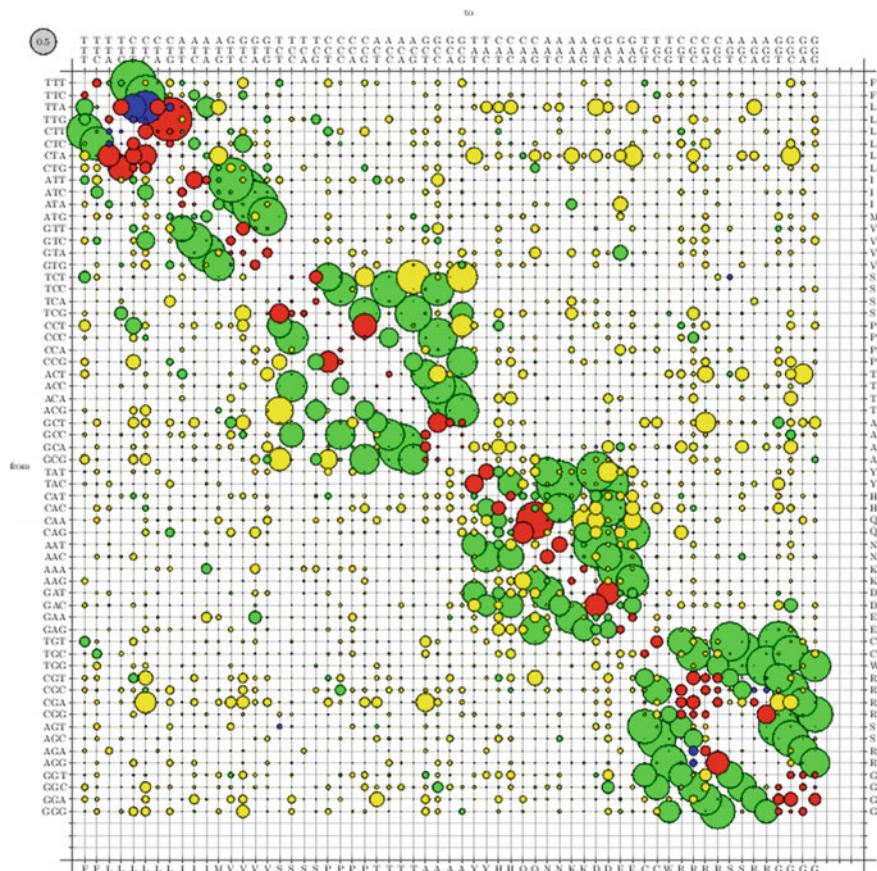


Fig. 21 Bubble plot of codon substitution rates computed with the help of XRate for orthologous group of *rbcL* genes for RuBisCo protein. The *rbcL* sequences from flowering land plants were taken from [268]. Fifty codon sequences were used for model generation. See Fig. 19 for color code and more details

8 Identification and Visualization of Selection Forces Acting on Coding Sequences

Identification of plausible signals of selection on protein-coding genes was and remains the main task of many codon-oriented approaches. This is an extremely challenging task because extant coding sequences are molded by a multitude of intertwined neutral and selective forces acting over different timescales and are sampled from poorly understood populations. This is a “technical” part of the explanation as to why there is no straightforward and universal procedure to detect selection on biological sequences. On a more conceptual level, there is great deal of chance and unknown in evolutionary trajectories, and these might not be appropriately framed

into the most thoroughly formulated models. These precautions notwithstanding, an overall consensus is that coding sequences carry the signals of selection and a methodologically careful analysis will reveal it. Here, we discuss what parameters derived from codon studies are the most insightful with regard to estimation of natural selection. Then, we will briefly describe the available tools that allow visualization of selective forces that act on coding sequences.

8.1 Molecular Evolution at the Codon Level: A Brief Introduction

Random mutations arising in an individual genome are a basic material for evolution. If a mutation increases reproductive success of the organism under given conditions (i.e., has “fitness” benefits [269]), it will spread in the population and become fixed faster. The above process of fixation of certain mutations with probability above random expectation is referred to as positive selection. Conversely, negative (purifying) selection eliminates detrimental mutations from the population, thereby conserving an existing amino acid sequence. A weak purifying selection that removes alleles with extreme deviations from the mean (optimal) fitness value creates the basis for stabilizing selection. The latter is thought to be the main force in phenotypic evolution, and recent works reinforce its importance during the molecular evolution as well [270].

Positive selection comes in several flavors depending on the timescale and the origin of the sequences analyzed. Particularly, the data may come from either a single population (e.g., mycobacteria isolated from the sputum of a patient) or different species. Positive selection that favors non-synonymous changes across interspecies sequences is known as diversifying selection. This type of selection can be assumed constant over a certain time frame, e.g., if it affects similarly a protein family or a set of related species. Within populations, positive selection can be either balancing or directional. Balancing selection increases the polymorphism level (when it confers fitness advantage to the carrier cell), while directional selection decreases it by driving the beneficial mutations to fixation. In the latter case, the pressure fades away as the mutation is no longer rare. Selective sweep is a result of positive selection within population where a fixed beneficial mutation reduces variation in the nearby (linked) loci. Generally, coding sequences from interspecies and population data pose different kinds of a problem for the detection of selective evolutionary forces. Genetic alterations that define interspecific markers occurred a long time ago and thus represent fixed mutations (substitutions). It is therefore relatively straightforward to analyze the selection across species. In contrast, intraspecific genetic variation is low and can be enriched for polymorphic sites (mutations not yet driven to fixation) segregating in the populations. This requires a proper adjustment of the analysis methods.

While beneficial and harmful mutations are under selective pressure, it appears that many mutations are neutral or nearly neutral with regard to their impact on fitness [271, 272]. Their maintenance in genes is determined by genetic drift and mutational pressure rather than selection. Synonymous codon substitutions were one of the earliest candidates for neutral mutations [273]. Current view of codon evolution portrays a complex picture where population structure and nonadaptive factors (such as mutation bias and genetic drift) play an important role alongside selection; an ongoing debate is about relative contributions of these forces in different cases and when the amount of mutations is enough to shape the gene's structure. Depending on an experimental model, an involvement of positive, purifying or stabilizing forces in shaping the codon usage has been suggested [67, 274], and different genome loci can experience a different kind and/or strength of selection [275]. No universal or taxon-specific rules seem to exist for codon sequence evolution. Therefore, researchers must carefully consider all possible evolutionary scenarios for the data at hand.

A single coding sequence cannot be used to pinpoint selection. Although high values of certain codon indices, such as CAI, ENC or Fop (see above), may imply that the gene is under selective pressure, it could be very difficult to tell its kind. Furthermore, most codon indices rely on reference information, and so the prediction is in fact based on more than one sequence. Codon volatility is the only notable attempt to infer selection from a single sequence [276]. The former can be defined as a probability that a point mutation in a codon leads to a non-synonymous codon. Each codon would therefore have its fixed volatility value, and gene volatility (ranging from 0.5 to 1.0) is an average value over all its codons. If a coding sequence consists of high volatility codons, then there is an increased probability that previous codon substitutions were non-synonymous. The volatility of codons can therefore be used as a statistic to reveal positive selection. The significance of gene volatility can be deduced from its comparison to the bootstrap distribution of volatility values of simulated sequences [277]. Numerous studies revealed no correlation between codon volatility and positive selection in a number of organisms, mounting strong critique of this approach as a way to detect selection [278–280]. Hence, at least two homologous coding sequences are needed to reveal the latter.

A number of methods for detecting intraspecific selection are based on the prediction of frequency of mutations (alleles) expected under the neutral regime of evolution within and among populations. If a genomic region has undergone selective sweep, then there would be low sequence diversity and an excess of rare mutations. Differential pressure of selection on populations would also leave a footprint in DNA sequences. These differences create a foundation for a suite of neutrality test statistics, such as Tajima's D , Fu's W , Wright's F_{ST} and others [281]. The problem with these neutrality tests is that their precision and interpretation depend on the demographic history of the population, which is often difficult to know.

Many approaches to detect selection on coding sequences are based on the analysis of non-synonymous (d_N) and synonymous (d_S) substitutions within pairwise or multiple alignments of homologous coding sequences. Assuming the neutral evolution as a null hypothesis, several neutrality tests were developed to detect selection that drives species divergence. For example, McDonalds–Kreitman test (Fig. 22) com-

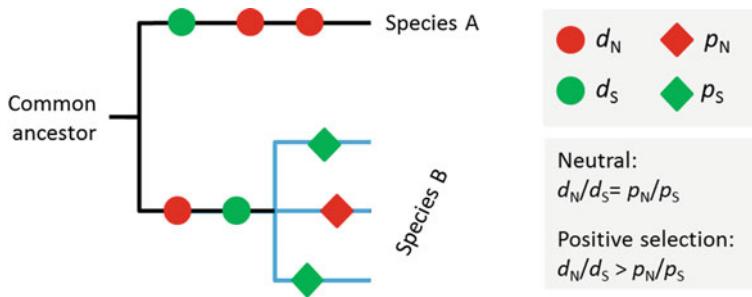


Fig. 22 McDonalds–Kreitman neutrality test. A toy phylogenetic tree features two clades, one represented by species A and another by three members of species B (blue lines). The test compares the non-synonymous/synonymous substitution ratio (d_N/d_S) on species branches (circles) to the non-synonymous/synonymous polymorphism ratio (p_N/p_S) on intraspecies lineages (diamonds). A graphical legend is shown to the right of the figure

pares the number of non-synonymous changes between and within species [282]. If this number is significantly higher between species, this is construed as a signal of diversifying selection.

The computation of d_N/d_S , or the ω -ratio, is also an established way to detect selection on sequences, which is most efficient when the sequences are derived from different species. Positive or negative selection is indicated when $\omega > 1$ and $\omega < 1$, respectively, whereas $\omega = 1$ suggests that sequences evolve neutrally. First methods to compute this ratio relied on counting of various changes in pairwise comparisons of codon sequences [233]. Counting approaches, however, suffer from two problems. First, they lack power when sequences are too similar or highly divergent and can be applied to pairs of sequences rather than to multiple sequence alignments. There is no proper framework to handle the uncertainty if the methods applied over a phylogeny in a pairwise fashion (although some solutions have been proposed on the basis of visual exploration of the model fit [283, 284]). Second, in most cases selection impacts only a few sites in a coding sequence, but counting methods have no means to provide estimate site-specific ω values [285]. Recent notable development of counting methods includes the renaissance counting, implemented in the package BEAST (see below). It maps substitutions throughout the phylogeny in the 4×4 nucleotide space, then counts d_N and d_S , their “neutral” expectations, and applies an empirical Bayes procedure to those counts to arrive at d_N/d_S estimates [286]. This approach is computationally efficient when it comes to the analysis of very large datasets and offers solutions to the aforementioned shortcomings of early counting methods.

Model-based methods of estimating the ω -ratio at least as good and more often outperform the counting methods. Even with only two sequences, maximum likelihood methods allow for accurate inferences [287]. Model-based approaches are naturally applicable to multiple sequence alignments, from which site-specific selection parameters can also be inferred. Unlike the counting approaches, maximum likelihood estimates have attractive statistical properties (e.g., convergence to the true

value as data size grows). It is desirable to use multiple sequence alignments instead of pairwise as the inferences will be more powerful, and selection can be studied with more sophisticated codon models as described above. A typical scheme is as follows. For a given dataset, infer a multiple codon sequence alignment and a phylogenetic tree (preferably using a codon model which can be done using the program **codonPhyML** [288]). Based on the inferred alignment and phylogeny, compare two codon models, one that allows selection (ω can be >1 for all or some sites and/or branches of the phylogeny) and another model that does not allow selection. Using statistical model comparisons, such as LRT (see above), one can deduce whether selection-permitting model describes the data better. The absence of differential selection pressure on synonymous and non-synonymous sites is an implicit condition of many ω -based approaches (which is not strictly obeyed; see preceding sections), although there are also models that allow variable d_N and d_S . If there is a reason to believe that not all synonymous sites are the same, this should be properly incorporated into modeling process. The ω -based approaches can also be applied to intraspecies (population) data; here, one should be careful about the initial assumptions on effective population sizes and mutation rates, as these could affect the conclusions. A detailed treatment of the use of parametric codon substitution models in natural selection studies is given in [1]. An entry-level tutorial on detecting selection in microbial genomes using ω -based methods can be found in [289].

8.2 Tools for Visual Representation of Selective Pressure on Protein-Coding Sequences

A number of software packages and online services are at the disposition of a researcher interested in the problem of detecting the positive selection in molecular data. Most of them are reviewed in [1]; here, we will mention the most popular as well as recently developed tools.

PAML is one of the first software packages for maximum likelihood phylogenetic analysis of nucleotide and amino acid sequences [290]. It includes the **CodeML** program that allows estimating parameters for a variety of codon models. Originally, PAML was available only as a command-line package; however, there is ongoing development of add-ons and wrappers that facilitate and visualize the results of PAML and CodeML in particular [291, 292].

HyPhy is an open-source phylogenetic package software that permits the inference of natural selection on the basis of ω estimated from multiple sequence alignments (DNA, codon, amino acid) as well as other data types, such as microsatellite counts [293]. A broad collection of complex models can be implemented in HyPhy, thus providing a great flexibility in analysis workflow. The companion **Datamonkey** 2.0 Web server provides a free access to most of the analysis options offered by HyPhy [294]. Datamonkey is also accessible through the **MEGA X** phylogenetic software package. Pervasive/episodic selection on sites, branches or entire genes can

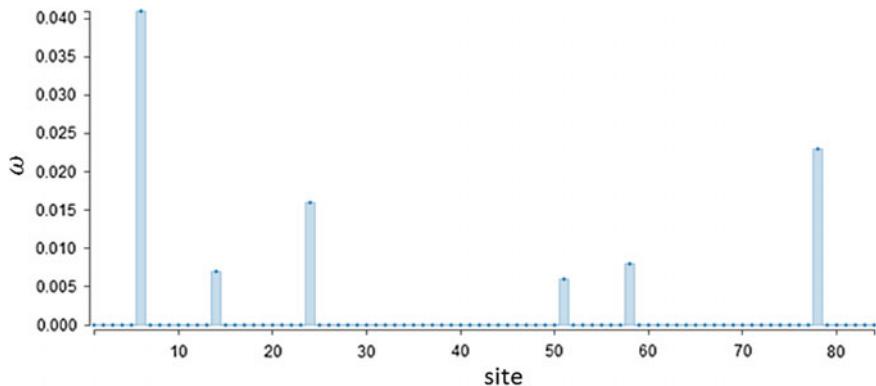


Fig. 23 Fixed effects likelihood (FEL) analysis of 56 sequences, 84 codons long, encoding AraC-type DNA binding domain of transcriptional factor AdpA from *Actinobacteria*. The branch leading to the micromonosporae was selected for FEL analysis. The *X*-axis represents codon positions, and *Y*-axis—omega value for each position. The ω values are much less than 1 (or zero if $d_N = 0$) for the chosen branch, indicating pervasive purifying selection (*p*-value threshold was set to 0.1)

be detected, either in the absence or in the presence of recombination; coevolution of positions of coding sequences can be estimated. The Web server reports the results in both tabular and graphical formats. The latter, in case of d_N and d_S values, is represented by a plot which shows the presumed sites under selection (Fig. 23).

BEAST 2.5 is an open-source, extensible software platform for Bayesian evolutionary analysis. Much like HyPhy, it allows to perform virtually any type of evolutionary analyses, both across and within species [295]. One particular feature of this package is the ability to integrate geographic maps and biological data, an important asset for those interested in visualizing complex spatial and demographic processes (e.g., epidemic outbreaks, etc.). **FRESCO** is a codon-based phylogenetic application for detection of viral genes excessively constrained in synonymous substitutions [296]. Due to the extremely dense organization of viral genomes, such unusual constraint may be an indicative of important functional role of given genome segment, such as secondary structure of RNA and packaging signals. **SELECTON** (<http://selecton.tau.ac.il/>) offers a set of codon models for Bayesian inference of positive and purifying selection [297]. The result of the inference is graphically displayed for each site using a color-coding scheme, as shown in Fig. 24.

9 Outlook

Current omics and systems biology approaches enable high-throughput readout of DNA sequences, levels of their transcription and translation, as well as the stability of respective mRNAs and proteins. The generated data provide ample evidence that protein-coding sequence controls not only the primary protein structure; through

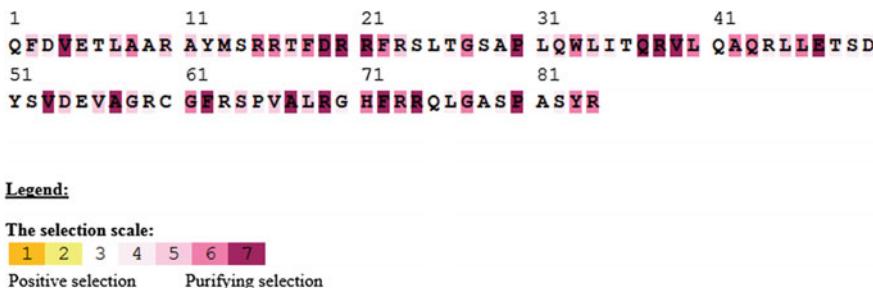


Fig. 24 Color-coded results of the SELECTON analysis of selective forces on the same dataset as the one used to prepare Fig. 23

modulation of translation speed, it may also determine mRNA stability and protein conformation. However, enormous complexity and scale of the molecular data make it nontrivial to understand the associated biological processes and to extract some general rules. Below, we attempt to summarize main issues in the area of codon usage and how visualization-oriented tools may improve the situation.

Computational and experimental approaches toward better understanding of intrinsic and context-dependent codon usage biases remain a fertile ground for database and software development. Traditional CUB indices enjoy sustained use, and new ones are constantly being reported. There is clear need for more extensive benchmarking studies of different CUB indices. This will help understand what they measure, how they are useful and what can be improved. There seem to be a script or program for virtually every aspect of CUB in biological software libraries, but not all of them are known or can be easily accessed by a wider community of biologists; some required tools are mentioned in respective section of this text. A more consistent action in this direction will undoubtedly deepen the insights into underlying biological phenomena. On a more fundamental level, CUB studies generate a flood of data that are quite complex and, often, a subject of disparate interpretation [298]. Much remains to be discovered in this area, and improved visualization may help to organize and embrace the complexities of codon usage biases.

Current CSMs largely neglect functional, structural or phenotypic data associated with the sequences under study. The evolution of coding sequences is ultimately dependent on the fitness effects of mutations, and these effects depend on the genetic background, starting from the intragenic level [299]. It is imperative to keep searching statistical and computational foundations so that CSMs can incorporate biochemical and structural information, or the genotype–phenotype effects. Mutation–selection framework is one example of how this can be accomplished [245], although there is a long way ahead before such models become a routine and reliable tool. As the sequencing data enter BigData arena, there is also a need to increase the computation speed of codon models. Likewise, there is a clear need for improved software engineering quality in case of programs for detecting selection [300], which will pay off in terms of their usage and gained knowledge.

Elaborating further details of sophisticated codon models is an important area of research activity. It is necessary to compare them under standardized conditions to fully reveal their capabilities. For example, recent studies suggest that the choice of the precise criteria for selection of nucleotide and amino acid substitution models has little influence on phylogeny or estimation of site-specific evolutionary rates. The most parameter-rich nucleotide model (general time reversible + $I + \Gamma$) fares well in the reconstruction [301], while simple Jukes-Cantor-type amino acid model adequately identifies rapidly evolving sites [302].

Huge number of annotated genomes offers unprecedented opportunities for modeling of coding sequences through empirical CSMs, e.g., monitoring evolving populations [249]. The latter are larger than nucleotide or amino acid substitution models, and they would especially benefit from visualization approaches, as bubble plots illustrate. The advantages of codon models for modeling and understanding protein evolution are clearly articulated in the specialized literature [1]; yet, the practical use of CSMs lags behind the simpler models. We believe that availability of Web-based applications to build and test empirical CSMs will help ameliorate this problem.

In a metaphorical way, the genetic code of an organism can be viewed as an interplay of “writers”—rules or forces that specify CUB patterns in the genome, and “readers”—components of the translational machinery, mainly tRNAs and ribosomes, that convert the code into proteins. Population genetic and natural selection theories provide a reliable conceptual framework to study “writers”; less is understood on the “readers” side. For example, there is growing layer of information on the roles of post-transcriptional tRNA modifications in regulating gene expression, which explains preferred codon usage in different biological systems [303, 304]. Incorporation of relevant biological knowledge into models of codon usage and evolution will improve their predictive power and utility. The picture is often worth a thousand words as it fuels and drives our ability to grasp the ideas, trends and associations. We therefore should not neglect the visualization aspect of codon models.

Acknowledgements B.O. thanks numerous students and coworkers who investigated various aspects of codon usage. Work in the laboratory of B.O. was supported by the grants from Ministry of Education and Science of Ukraine and State Fund for Fundamental Research. M.A. thanks the Swiss National Science Foundation for research funding (grant 31003A_182330/1).

References

1. Cannarozzi GM, Schneider A (eds) (2012) Codon evolution. Mechanisms and models. Oxford University Press, New York, 297 p. ISBN 978–0–19–960116–5
2. Margulies M, Egholm M, Altman WE, Attiya S, Bader JS, Bemben LA, Berka J, Braverman MS, Chen YJ, Chen Z, Dewell SB, Du L, Fierro JM, Gomes XV, Godwin BC, He W, Helgesen S, Ho CH, Irzyk GP, Jando SC, Alenquer ML, Jarvie TP, Jirage KB, Kim JB, Knight JR, Lanza JR, Leamon JH, Lefkowitz SM, Lei M, Li J, Lohman KL, Lu H, Makhijani VB, McDade KE, McKenna MP, Myers EW, Nickerson E, Nobile JR, Plant R, Puc BP, Ronan MT, Roth

- GT, Sarkis GJ, Simons JF, Simpson JW, Srinivasan M, Tartaro KR, Tomasz A, Vogt KA, Volkmer GA, Wang SH, Wang Y, Weiner MP, Yu P, Begley RF, Rothberg JM (2005) Genome sequencing in microfabricated high-density picolitre reactors. *Nature* 437(7057):376–380
- 3. Rothberg JM, Leamon JH (2008) The development and impact of 454 sequencing. *Nat Biotechnol* 26(10):1117–1124. <https://doi.org/10.1038/nbt1485>
 - 4. Slatko BE, Gardner AF, Ausubel FM (2018) Overview of next-generation sequencing technologies. *Curr Protoc Mol Biol* 122(1):e59. <https://doi.org/10.1002/cpmb.59>
 - 5. O'Donoghue SI, Baldi BF, Clark SJ, Darling AE, Hogan JM, Kaur S, Maier-Hein L, McCarthy DJ, Moore WJ, Stenau E, Swedlow JR, Vuong J, Procter JB (2018) Visualization of biomedical data. *Annu Rev Biomed Data Sci* 1:275–304. <https://doi.org/10.1146/annurev-biodatasci-080917-013424>
 - 6. Liu X, Zhang J, Ni F, Dong X, Han B, Han D, Ji Z, Zhao Y (2010) Genome wide exploration of the origin and evolution of amino acids. *BMC Evol Biol* 15(10):77. <https://doi.org/10.1186/1471-2148-10-77>
 - 7. Jordan IK, Kondrashov FA, Adzhubei IA, Wolf YI, Koonin EV, Kondrashov AS, Sunyaev S (2005) A universal trend of amino acid gain and loss in protein evolution. *Nature* 433(7026):633–638
 - 8. Fimmel E, Strüngmann L (2018) Mathematical fundamentals for the noise immunity of the genetic code. *Biosystems* 164:186–198. <https://doi.org/10.1016/j.biosystems.2017.09.007>
 - 9. Keeling PJ (2016) Genomics: evolution of the genetic code. *Curr Biol* 26(18):R851–R853. <https://doi.org/10.1016/j.cub.2016.08.005>
 - 10. Koonin EV, Novozhilov AS (2017) Origin and evolution of the universal genetic code. *Annu Rev Genet* 27(51):45–62. <https://doi.org/10.1146/annurev-genet-120116-024713>
 - 11. Heaphy SM, Mariotti M, Gladyshev VN, Atkins JF, Baranov PV (2016) Novel ciliate genetic code variants including the reassignment of all three stop codons to sense codons in *Condylostoma magnum*. *Mol Biol Evol* 33(11):2885–2899
 - 12. Mühlhausen S, Schmitt HD, Pan KT, Plessmann U, Urlaub H, Hurst LD, Kollmar M (2018) Endogenous stochastic decoding of the CUG codon by competing Ser- and Leu-tRNAs in *Ascoidea asiatica*. *Curr Biol* 28(13):2046–2057.e5. <https://doi.org/10.1016/j.cub.2018.04.085>
 - 13. Miranda I, Rocha R, Santos MC, Mateus DD, Moura GR, Carreto L, Santos MA (2007) A genetic code alteration is a phenotype diversity generator in the human pathogen *Candida albicans*. *PLoS ONE* 2(10):e996
 - 14. Väre VY, Eruysal ER, Narendran A, Sarachan KL, Agris PF (2011) Chemical and conformational diversity of modified nucleosides affects tRNA structure and function. *Biomolecules* 7(1):pii: E29. <https://doi.org/10.3390/biom7010029>
 - 15. Agris PF, Narendran A, Sarachan K, Väre VYP, Eruysal E (2017) The importance of being modified: the role of RNA modifications in translational fidelity. *Enzymes* 41:1–50. <https://doi.org/10.1016/bs.enz.2017.03.005>
 - 16. Schweizer U, Bohleber S, Fradejas-Villar N (2017) The modified base isopentenyladenosine and its derivatives in tRNA. *RNA Biol* 14(9):1197–1208. <https://doi.org/10.1080/15476286.2017.1294309>
 - 17. Hori H (2017) Transfer RNA methyltransferases with a SpoU-TrmD (SPOUT) fold and their modified nucleosides in tRNA. *Biomolecules* 7(1):pii: E23. <https://doi.org/10.3390/biom7010023>
 - 18. Hou YM, Masuda I, Gamper H (2019) Codon-Specific Translation by m(1)G37 Methylation of tRNA. *Front Genet* 10(9):713. <https://doi.org/10.3389/fgene.2018.00713>
 - 19. Pan T (2018) Modifications and functional genomics of human transfer RNA. *Cell Res* 28(4):395–404. <https://doi.org/10.1038/s41422-018-0013-y>
 - 20. Schimmel P (2018) The emerging complexity of the tRNA world: mammalian tRNAs beyond protein synthesis. *Nat Rev Mol Cell Biol* 19(1):45–58. <https://doi.org/10.1038/nrm.2017.77>
 - 21. Silva RM, Paredes JA, Moura GR, Manadas B, Lima-Costa T, Rocha R, Miranda I, Gomes AC, Koerkamp MJ, Perrot M, Holstege FC, Boucherie H, Santos MA (2007) Critical roles for a genetic code alteration in the evolution of the genus *Candida*. *EMBO J* 26(21):4555–4565

22. Zhang Z, Yu J (2011) On the organizational dynamics of the genetic code. *Genomics Proteomics Bioinformatics*. 9(1–2):21–29. [https://doi.org/10.1016/S1672-0229\(11\)60004-1](https://doi.org/10.1016/S1672-0229(11)60004-1)
23. Rosandić M, Paar V (2014) Codon sextets with leading role of serine create “ideal” symmetry classification scheme of the genetic code. *Gene* 543(1):45–52. <https://doi.org/10.1016/j.gene.2014.04.009>
24. José MV, Zamudio GS, Morgado ER (2017) A unified model of the standard genetic code. *R Soc Open Sci* 4(3):160908. <https://doi.org/10.1098/rsos.160908>
25. Acevedo-Rocha CG, Budisa N (2016) Xenomicobiology: a roadmap for genetic code engineering. *Microb Biotechnol* 9(5):666–676. <https://doi.org/10.1111/1751-7915.12398>
26. van der Gulik PT, Hoff WD (2016) Anticodon modifications in the tRNA set of LUCA and the fundamental regularity in the standard genetic code. *PLoS ONE* 11(7):e0158342. <https://doi.org/10.1371/journal.pone.0158342>
27. Grosjean H, Westhof E (2016) An integrated, structure- and energy-based view of the genetic code. *Nucleic Acids Res* 44(17):8020–8040. <https://doi.org/10.1093/nar/gkw608>
28. Subramaniam AR, Pan T, Cluzel P (2013) Environmental perturbations lift the degeneracy of the genetic code to regulate protein levels in bacteria. *Proc Natl Acad Sci U S A* 110(6):2419–2424. <https://doi.org/10.1073/pnas.1211077110>
29. Moukadir I, Garzón MJ, Björk GR, Armengod ME (2014) The output of the tRNA modification pathways controlled by the *Escherichia coli* MnmEG and MnmC enzymes depends on the growth conditions and the tRNA species. *Nucleic Acids Res* 42(4):2602–2623. <https://doi.org/10.1093/nar/gkt1228>
30. Asano K, Suzuki T, Saito A, Wei FY, Ikeuchi Y, Numata T, Tanaka R, Yamane Y, Yamamoto T, Goto T, Kishita Y, Murayama K, Ohtake A, Okazaki Y, Tomizawa K, Sakaguchi Y, Suzuki T (2018) Metabolic and chemical regulation of tRNA modification associated with taurine deficiency and human disease. *Nucleic Acids Res* 46(4):1565–1583. <https://doi.org/10.1093/nar/gky068>
31. Kirchner S, Ignatova Z (2015) Emerging roles of tRNA in adaptive translation, signalling dynamics and disease. *Nat Rev Genet* 16(2):98–112. <https://doi.org/10.1038/nrg3861>
32. Rogers SO (2019) Evolution of the genetic code based on conservative changes of codons, amino acids, and aminoacyl tRNA synthetases. *J Theor Biol* 7(466):1–10. <https://doi.org/10.1016/j.jtbi.2019.01.022>
33. Itzkovitz S, Alon U (2007) The genetic code is nearly optimal for allowing additional information within protein-coding sequences. *Genome Res* 17(4):405–412
34. Itzkovitz S, Hodis E, Segal E (2010) Overlapping codes within protein-coding sequences. *Genome Res* 20(11):1582–1589. <https://doi.org/10.1101/gr.105072.110>
35. Bollenbach T, Vetsigian K, Kishony R (2007) Evolution and multilevel optimization of the genetic code. *Genome Res* 17(4):401–404
36. Wnętrzak M, Błażej P, Mackiewicz D, Mackiewicz P (2018) The optimality of the standard genetic code assessed by an eight-objective evolutionary algorithm. *BMC Evol Biol* 18(1):192. <https://doi.org/10.1186/s12862-018-1304-0>
37. Błażej P, Wnętrzak M, Mackiewicz D, Gagat P, Mackiewicz P (2019) Many alternative and theoretical genetic codes are more robust to amino acid replacements than the standard genetic code. *J Theor Biol* 7(464):21–32. <https://doi.org/10.1016/j.jtbi.2018.12.030>
38. Kuruoglu EE, Arndt PF (2017) The information capacity of the genetic code: is the natural code optimal? *J Theor Biol* 21(419):227–237. <https://doi.org/10.1016/j.jtbi.2017.01.046>
39. Agarwal D, Gregory ST, O’Connor M (2011) Error-prone and error-restrictive mutations affecting ribosomal protein S12. *J Mol Biol* 410(1):1–9. <https://doi.org/10.1016/j.jmb.2011.04.068>
40. Robinson LJ, Cameron AD, Stavrinides J (2015) Spontaneous and on point: do spontaneous mutations used for laboratory experiments cause pleiotropic effects that might confound bacterial infection and evolution assays? *FEMS Microbiol Lett* 362(21):pii: fnv177. <https://doi.org/10.1093/femsle/fnv177>
41. An W, Chin JW (2011) Orthogonal gene expression in *Escherichia coli*. *Methods Enzymol* 497:115–134. <https://doi.org/10.1016/B978-0-12-385075-1.00005-6>

42. Liu CC, Jewett MC, Chin JW, Voigt CA (2018) Toward an orthogonal central dogma. *Nat Chem Biol* 14(2):103–106. <https://doi.org/10.1038/nchembio.2554>
43. Ishikawa J, Hotta K (1999) FramePlot: a new implementation of the frame analysis for predicting protein-coding regions in bacterial DNA with a high G+C content. *FEMS Microbiol Lett* 174(2):251–253
44. Fickett JW, Tung CS (1992) Assessment of protein coding measures. *Nucleic Acids Res* 20(24):6441–6450
45. Azad RK, Borodovsky M (2004) Probabilistic methods of identifying genes in prokaryotic genomes: connections to the HMM theory. *Brief Bioinform* 5(2):118–130
46. Pride DT, Meinersmann RJ, Wassenaar TM, Blaser MJ (2003) Evolutionary implications of microbial genome tetranucleotide frequency biases. *Genome Res* 13(2):145–158
47. Teeling H, Waldmann J, Lombardot T, Bauer M, Glöckner FO (2004) TETRA: a web-service and a stand-alone program for the analysis and comparison of tetranucleotide usage patterns in DNA sequences. *BMC Bioinform* 26(5):163
48. Richter M, Rosselló-Móra R, Oliver Glöckner F, Peplies J (2016) JSpeciesWS: a web server for prokaryotic species circumscription based on pairwise genome comparison. *Bioinformatics* 32(6):929–931. <https://doi.org/10.1093/bioinformatics/btv681>
49. Wang Y, Zeng Z, Liu TL, Sun L, Yao Q, Chen KP (2019) TA, GT and AC are significantly under-represented in open reading frames of prokaryotic and eukaryotic protein-coding genes. *Mol Genet Genomics.* <https://doi.org/10.1007/s00438-019-01535-1>
50. Akogwu I, Wang N, Zhang C, Gong P (2016) A comparative study of k-spectrum-based error correction methods for next-generation sequencing data analysis. *Hum Genomics* 10(Suppl 2):20. <https://doi.org/10.1186/s40246-016-0068-0>
51. Mapleson D, Garcia Accinelli G, Kettleborough G, Wright J, Clavijo BJ (2017) KAT: a K-mer analysis toolkit to quality control NGS datasets and genome assemblies. *Bioinformatics* 33(4):574–576. <https://doi.org/10.1093/bioinformatics/btw663>
52. Sheppard SK, Guttman DS, Fitzgerald JR (2018) Population genomics of bacterial host adaptation. *Nat Rev Genet* 19(9):549–565. <https://doi.org/10.1038/s41576-018-0032-z>
53. Camiolo S, Porceddu A (2018) corseq: fast and efficient identification of favoured codons from next generation sequencing reads. *PeerJ.* 4(6):e5099. <https://doi.org/10.7717/peerj.5099>
54. Lees JA, Vehkala M, Välimäki N, Harris SR, Chewapreecha C, Croucher NJ, Marttinen P, Davies MR, Steer AC, Tong SY, Honkela A, Parkhill J, Bentley SD, Corander J (2016) Sequence element enrichment analysis to determine the genetic basis of bacterial phenotypes. *Nat Commun* 16(7):12797. <https://doi.org/10.1038/ncomms12797>
55. Mohamadi H, Khan H, Birol I (2017) ntCard: a streaming algorithm for cardinality estimation in genomics data. *Bioinformatics* 33(9):1324–1330. <https://doi.org/10.1093/bioinformatics/btw832>
56. Manekar SC, Sathe SR (2018) A benchmark study of k-mer counting methods for high-throughput sequencing. *Gigascience* 7(12). <https://doi.org/10.1093/gigascience/giy125>
57. Fuglsang A (2004) Nucleotides downstream of start codons show marked non-randomness in *Escherichia coli* but not in *Bacillus subtilis*. *Antonie Van Leeuwenhoek* 86(2):149–158
58. Rokytksky I, Kulaha S, Mutenko H, Rabyk M, Ostash B (2017) Peculiarities of codon context and substitution within streptomycete genomes. *Vsnn Lviv Univ Ser Biol* 75:66–74. <https://doi.org/10.30970/vlubs.2017.75.07>
59. Knight RD, Freeland SJ, Landweber LF (2001) A simple model based on mutation and selection explains trends in codon and amino-acid usage and GC composition within and across genomes. *Genome Biol* 2(4):RESEARCH0010
60. Ikemura T (1985) Codon usage and tRNA content in unicellular and multicellular organisms. *Mol Biol Evol* 2(1):13–34
61. Higgs PG, Ran W (2008) Coevolution of codon usage and tRNA genes leads to alternative stable states of biased codon usage. *Mol Biol Evol* 25(11):2279–2291. <https://doi.org/10.1093/molbev/msn173>
62. Kanaya S, Yamada Y, Kinouchi M, Kudo Y, Ikemura T (2001) Codon usage and tRNA genes in eukaryotes: correlation of codon usage diversity with translation efficiency and with CG-dinucleotide usage as assessed by multivariate analysis. *J Mol Evol* 53(4–5):290–298

63. Yu C-H, Dang Y, Zhou Z et al (2015) Codon usage influences the local rate of translation elongation to regulate co-translational protein folding. *Mol Cell* 59:744–754
64. Quax TE, Claassens NJ, Söll D, van der Oost J (2015) Codon bias as a means to fine-tune gene expression. *Mol Cell* 59(2):149–161. <https://doi.org/10.1016/j.molcel.2015.05.035>
65. Ikemura T (1981) Correlation between the abundance of *Escherichia coli* transfer RNAs and the occurrence of the respective codons in its protein genes: a proposal for a synonymous codon choice that is optimal for the *E. coli* translational system. *J Mol Biol* 151(3):389–409
66. dos Reis M, Savva R, Wernisch L (2004) Solving the riddle of codon usage preferences: a test for translational selection. *Nucleic Acids Res* 32(17):5036–5044
67. Hershberg R, Petrov DA (2008) Selection on codon bias. *Annu Rev Genet* 42:287–299. <https://doi.org/10.1146/annurev.genet.42.110807.091442>
68. Gribskov M, Devereux J, Burgess RR (1984) The codon preference plot: graphic analysis of protein coding sequences and prediction of gene expression. *Nucleic Acids Res* 12(1 Pt 2):539–549
69. Garcia-Vallve S, Guzman E, Montero MA, Romeu A (2003) HGT-DB: a database of putative horizontally transferred genes in prokaryotic complete genomes. *Nucleic Acids Res* 31(1):187–189
70. Puigbò P, Romeu A, Garcia-Vallvé S (2008) HEG-DB: a database of predicted highly expressed genes in prokaryotic complete genomes under translational selection. *Nucleic Acids Res* 36(Database issue):D524–D527
71. Paulet D, David A, Rivals E (2017) Ribo-seq enlightens codon usage bias. *DNA Res* 24(3):303–2100. <https://doi.org/10.1093/dnares/dsw062>
72. Sharp PM, Li WH (1987) The codon Adaptation Index—a measure of directional synonymous codon usage bias, and its potential applications. *Nucleic Acids Res* 15(3):1281–1295
73. Dittmar KA, Sørensen MA, Elf J, Ehrenberg M, Pan T (2005) Selective charging of tRNA isoacceptors induced by amino-acid starvation. *EMBO Rep* 6(2):151–157
74. Welch M, Govindarajan S, Ness JE, Villalobos A, Gurney A, Minshull J, Gustafsson C (2009) Design parameters to control synthetic gene expression in *Escherichia coli*. *PLoS ONE* 4(9):e7002. <https://doi.org/10.1371/journal.pone.0007002>
75. Wu GD, Chen J, Hoffmann C, Bittinger K, Chen YY, Keilbaugh SA, Bewtra M, Knights D, Walters WA, Knight R, Sinha R, Gilroy E, Gupta K, Baldassano R, Nessel L, Li H, Bushman FD, Lewis JD (2011) Linking long-term dietary patterns with gut microbial enterotypes. *Science* 334(6052):105–108. <https://doi.org/10.1126/science.1208344>
76. Koropatkin NM, Cameron EA, Martens EC (2012) How glycan metabolism shapes the human gut microbiota. *Nat Rev Microbiol* 10(5):323–335. <https://doi.org/10.1038/nrmicro2746>
77. Hodgson DA (2000) Primary metabolism and its control in streptomycetes: a most unusual group of bacteria. *Adv Microb Physiol* 42:47–238
78. Ho A, Di Lonardo DP, Bodelier PL (2017) Revisiting life strategy concepts in environmental microbial ecology. *FEMS Microbiol Ecol* 93(3). <https://doi.org/10.1093/femsec/fix006>
79. Nakao A, Yoshihama M, Kenmochi N (2004) RPG: the Ribosomal Protein Gene database. *Nucleic Acids Res* 32(Database issue):D168–D170
80. Carbone A, Zinovyev A, Képès F (2003) Codon adaptation index as a measure of dominating codon bias. *Bioinformatics* 19(16):2005–2015
81. Raiford DW, Doom TE, Krane DE, Raymer ME (2011) A genetic optimization approach for isolating translational efficiency bias. *IEEE/ACM Trans Comput Biol Bioinf* 8(2):342–352
82. Xia X (2015) A major controversy in codon-anticodon adaptation resolved by a new codon usage index. *Genetics* 199(2):573–579. <https://doi.org/10.1534/genetics.114.172106>
83. Kudla G, Murray AW, Tollervey D, Plotkin JB (2009) Coding-sequence determinants of gene expression in *Escherichia coli*. *Science* 324(5924):255–258. <https://doi.org/10.1126/science.1170160>
84. Garcia V, Zoller S, Anisimova M (2018) Accounting for programmed ribosomal frameshifting in the computation of codon usage bias indices. *G3 (Bethesda)* 8(10):3173–3183. <https://doi.org/10.1534/g3.118.200185>

85. Wei Y, Silke JR, Xia X (2019) An improved estimation of tRNA expression to better elucidate the coevolution between tRNA abundance and codon usage in bacteria. *Sci Rep* 9(1):3184. <https://doi.org/10.1038/s41598-019-39369-x>
86. Pechmann S, Frydman J (2013) Evolutionary conservation of codon optimality reveals hidden signatures of cotranslational folding. *Nat Struct Mol Biol* 20(2):237–243. <https://doi.org/10.1038/nsmb.2466>
87. Sabi R, Tuller T (2014) Modelling the efficiency of codon-tRNA interactions based on codon usage bias. *DNA Res* 21(5):511–526. <https://doi.org/10.1093/dnares/dsu017>
88. Roymondal U, Das S, Sahoo S (2009) Predicting gene expression level from relative codon usage bias: an application to *Escherichia coli* genome. *DNA Res* 16(1):13–30. <https://doi.org/10.1093/dnares/dsn029>
89. Wright F (1990) The ‘effective number of codons’ used in a gene. *Gene* 87(1):23–29
90. Fuglsang A (2004) The ‘effective number of codons’ revisited. *Biochem Biophys Res Commun* 317(3):957–964
91. Novembre JA (2002) Accounting for background nucleotide composition when measuring codon usage bias. *Mol Biol Evol* 19(8):1390–1394
92. Liu X (2013) A more accurate relationship between ‘effective number of codons’ and GC3s under assumptions of no selection. *Comput Biol Chem* 42:35–39. <https://doi.org/10.1016/j.combiolchem.2012.11.003>
93. Sun X, Yang Q, Xia X (2013) An improved implementation of effective number of codons (N_c). *Mol Biol Evol* 30(1):191–196. <https://doi.org/10.1093/molbev/mss201>
94. Liu SS, Hockenberry AJ, Jewett MC, Amaral LAN (2018) A novel framework for evaluating the performance of codon usage bias metrics. *JR Soc Interface* 15(138):pii: 20170667. <https://doi.org/10.1098/rsif.2017.0667>
95. Zhang Z, Li J, Cui P, Ding F, Li A, Townsend JP, Yu J (2012) Codon Deviation Coefficient: a novel measure for estimating codon usage bias and its statistical significance. *BMC Bioinform* 22(13):43. <https://doi.org/10.1186/1471-2105-13-43>
96. Gilchrist MA, Shah P, Zaretzki R (2009) Measuring and detecting molecular adaptation in codon usage against nonsense errors during protein translation. *Genetics* 183(4):1493–1505. <https://doi.org/10.1534/genetics.109.108209>
97. Chou T (2003) Ribosome recycling, diffusion, and mRNA loop formation in translational regulation. *Biophys J* 85(2):755–773
98. Mitarai N, Sneppen K, Pedersen S (2008) Ribosome collisions and translation efficiency: optimization by codon usage and mRNA destabilization. *J Mol Biol* 382(1):236–245. <https://doi.org/10.1016/j.jmb.2008.06.068>
99. Gilchrist MA, Chen WC, Shah P, Landerer CL, Zaretzki R (2015) Estimating gene expression and codon-specific translational efficiencies, mutation biases, and selection coefficients from genomic data alone. *Genome Biol Evol* 7(6):1559–1579. <https://doi.org/10.1093/gbe/evv087>
100. Proshkin S, Rahmouni AR, Mironov A, Nudler E (2010) Cooperation between translating ribosomes and RNA polymerase in transcription elongation. *Science* 328(5977):504–508. <https://doi.org/10.1126/science.1184939>
101. Boël G, Letso R, Neely H, Price WN, Wong KH, Su M, Luff J, Valecha M, Everett JK, Acton TB, Xiao R, Montelione GT, Aalberts DP, Hunt JF (2016) Codon influence on protein expression in *E. coli* correlates with mRNA levels. *Nature* 529(7586):358–363. <https://doi.org/10.1038/nature16509>
102. Bellaousov S, Reuter JS, Seetin MG, Mathews DH (2013) RNAsstructure: web servers for RNA secondary structure prediction and analysis. *Nucleic Acids Res* 41(Web Server issue):W471–W474. <https://doi.org/10.1093/nar/gkt290>
103. Bentele K, Saffert P, Rauscher R, Ignatova Z, Blüthgen N (2013) Efficient translation initiation dictates codon usage at gene start. *Mol Syst Biol* 18(9):675. <https://doi.org/10.1038/msb.2013.32>
104. Kelsic ED, Chung H, Cohen N, Park J, Wang HH, Kishony R (2016) RNA structural determinants of optimal codons revealed by MAGE-Seq. *Cell Syst*. 3(6):563–571.e6. <https://doi.org/10.1016/j.cels.2016.11.004>

105. Frumkin I, Schirman D, Rotman A, Li F, Zahavi L, Mordret E, Asraf O, Wu S, Levy SF, Pilpel Y (2017) Gene architectures that minimize cost of gene expression. *Mol Cell* 65(1):142–153. <https://doi.org/10.1016/j.molcel.2016.11.007>
106. Hanson G, Alhusaini N, Morris N, Sweet T, Coller J (2018) Translation elongation and mRNA stability are coupled through the ribosomal A-site. *RNA* 24(10):1377–1389. <https://doi.org/10.1261/rna.066787.118>
107. Arango D, Sturgill D, Alhusaini N, Dillman AA, Sweet TJ, Hanson G, Hosogane M, Sinclair WR, Nanan KK, Mandler MD, Fox SD, Zenguya TT, Andresson T, Meier JL, Coller J, Oberdoerffer S (2018) Acetylation of cytidine in mRNA promotes translation efficiency. *Cell* 175(7):1872–1886.e24. <https://doi.org/10.1016/j.cell.2018.10.030>
108. Schikora-Tamarit MÀ, Carey LB (2018) Poor codon optimality as a signal to degrade transcripts with frameshifts. *Transcription* 9(5):327–333. <https://doi.org/10.1080/21541264.2018.1511676>
109. Lykke-Andersen S, Jensen TH (2015) Nonsense-mediated mRNA decay: an intricate machinery that shapes transcriptomes. *Nat Rev Mol Cell Biol* 16(11):665–677. <https://doi.org/10.1038/nrm4063>
110. Radhakrishnan A, Chen YH, Martin S, Alhusaini N, Green R, Coller J (2016) The DEAD-box protein Dhh1p couples mRNA decay and translation by monitoring codon optimality. *Cell* 167(1):122–132.e9. <https://doi.org/10.1016/j.cell.2016.08.053>
111. Presnyak V, Alhusaini N, Chen YH, Martin S, Morris N, Kline N, Olson S, Weinberg D, Baker KE, Graveley BR, Coller J (2015) Codon optimality is a major determinant of mRNA stability. *Cell* 160(6):1111–1124. <https://doi.org/10.1016/j.cell.2015.02.029>
112. Carneiro RL, Requião RD, Rossetto S, Domitrovic T, Palhano FL (2019) Codon stabilization coefficient as a metric to gain insights into mRNA stability and codon bias and their relationships with translation. *Nucleic Acids Res* 47(5):2216–2228. <https://doi.org/10.1093/nar/gkz033>
113. Kumar S, Stecher G, Li M, Knyaz C, Tamura K (2018) MEGA X: molecular evolutionary genetics analysis across computing platforms. *Mol Biol Evol* 35(6):1547–1549. <https://doi.org/10.1093/molbev/msy096>
114. Xia X (2018) DAMBE7: new and improved tools for data analysis in molecular biology and evolution. *Mol Biol Evol* 35(6):1550–1552. <https://doi.org/10.1093/molbev/msy073>
115. Supek F, Vlahovicek K (2004) INCA: synonymous codon usage analysis and clustering by means of self-organizing map. *Bioinformatics* 20(14):2329–2330
116. Peden JF (2005) CodonW, p. 1. <https://sourceforge.net/projects/codonw/>. Last accessed Apr 2019
117. Vetrivel U, Arunkumar V, Dorairaj S (2007) ACUA: a software tool for automated codon usage analysis. *Bioinformation* 2(2):62–63
118. Angelotti MC, Bhuiyan SB, Chen G, Wan XF (2007) CodonO: codon usage bias analysis within and across genomes. *Nucleic Acids Res* 35(Web Server issue):W132–W136
119. Miller JB, Brase LR, Ridge PG (2019) ExtRamp: a novel algorithm for extracting the ramp sequence based on the tRNA adaptation index or relative codon adaptiveness. *Nucleic Acids Res.* <https://doi.org/10.1093/nar/gky1193>
120. Tuller T, Carmi A, Vestigian K, Navon S, Dorfan Y, Zaborske J, Pan T, Dahan O, Furman I, Pilpel Y (2010) An evolutionarily conserved mechanism for controlling the efficiency of protein translation. *Cell* 141(2):344–354. <https://doi.org/10.1016/j.cell.2010.03.031>
121. Wu G, Culley DE, Zhang W (2005) Predicted highly expressed genes in the genomes of *Streptomyces coelicolor* and *Streptomyces avermitilis* and the implications for their metabolism. *Microbiology* 151(Pt 7):2175–2187
122. Grote A, Hiller K, Scheer M, Münch R, Nörtemann B, Hempel DC, Jahn D (2005) JCcat: a novel tool to adapt codon usage of a target gene to its potential expression host. *Nucleic Acids Res* 33(Web Server issue):W526–W531
123. Puigbò P, Bravo IG, Garcia-Vallve S (2008) CAIcal: a combined set of tools to assess codon usage adaptation. *Biol Direct* 16(3):38. <https://doi.org/10.1186/1745-6150-3-38>
124. McInerney JO (1998) GCUA: general codon usage analysis. *Bioinformatics* 14(4):372–373

125. Sabi R, Volfovitch Daniel R, Tuller T (2017) stAIcalc: tRNA adaptation index calculator based on species-specific weights. *Bioinformatics* 33(4):589–591. <https://doi.org/10.1093/bioinformatics/btw647>
126. Athey J, Alexaki A, Osipova E, Rostovtsev A, Santana-Quintero LV, Katneni U, Simonyan V, Kimchi-Sarfaty C (2017) A new and updated resource for codon usage tables. *BMC Bioinform* 18(1):391. <https://doi.org/10.1186/s12859-017-1793-7>
127. Yoon J, Chung YJ, Lee M (2018) STADIUM: species-specific tRNA adaptive index compendium. *Genomics Inform* 16(4):e28. <https://doi.org/10.5808/GI.2018.16.4.e28>
128. Plotkin JB, Kudla G (2011) Synonymous but not the same: the causes and consequences of codon bias. *Nat Rev Genet* 12(1):32–42. <https://doi.org/10.1038/nrg2899>
129. Ghaemmaghami S, Huh WK, Bower K, Howson RW, Belle A, Dephoure N, O’Shea EK, Weissman JS (2003) Global analysis of protein expression in yeast. *Nature* 425(6959):737–741
130. Ishihama Y, Schmidt T, Rappaport J, Mann M, Hartl FU, Kerner MJ, Frishman D (2008) Protein abundance profiling of the *Escherichia coli* cytosol. *BMC Genom* 27(9):102. <https://doi.org/10.1186/1471-2164-9-102>
131. Liu Y, Beyer A, Aebersold R (2016) On the dependency of cellular protein levels on mRNA abundance. *Cell* 165(3):535–550. <https://doi.org/10.1016/j.cell.2016.03.014>
132. Hanson G, Coller J (2018) Codon optimality, bias and usage in translation and mRNA decay. *Nat Rev Mol Cell Biol* 19(1):20–30. <https://doi.org/10.1038/nrm.2017.91>
133. Frumkin I, Lajoie MJ, Gregg CJ, Hornung G, Church GM, Pilpel Y (2018) Codon usage of highly expressed genes affects proteome-wide translation efficiency. *Proc Natl Acad Sci U S A* 115(21):E4940–E4949. <https://doi.org/10.1073/pnas.1719375115>
134. Puigbò P, Guzmán E, Romeu A, Garcia-Vallvé S (2007) OPTIMIZER: a web server for optimizing the codon usage of DNA sequences. *Nucleic Acids Res* 35(Web Server issue):W126–W131
135. Hatfield GW, Roth DA (2007) Optimizing scaleup yield for protein production: computationally optimized DNA assembly (CODA) and translation engineering. *Biotechnol Annu Rev* 13:27–42
136. Cheng BYH, Nogales A, de la Torre JC, Martínez-Sobrido L (2017) Development of live-attenuated arenavirus vaccines based on codon deoptimization of the viral glycoprotein. *Virology* 515(501):35–46. <https://doi.org/10.1016/j.virol.2016.11.001>
137. Jia W, Higgs PG (2008) Codon usage in mitochondrial genomes: distinguishing context-dependent mutation from translational selection. *Mol Biol Evol* 25(2):339–351
138. Aalberts DP, Boël G, Hunt JF (2017) Codon clarity or conundrum? *Cell Syst.* 4(1):16–19. <https://doi.org/10.1016/j.cels.2017.01.004>
139. Webster GR, Teh AY, Ma JK (2017) Synthetic gene design-The rationale for codon optimization and implications for molecular pharming in plants. *Biotechnol Bioeng* 114(3):492–502. <https://doi.org/10.1002/bit.26183>
140. Mauro VP, Chappell SA (2018) Considerations in the use of codon optimization for recombinant protein expression. *Methods Mol Biol* 1850:275–288. https://doi.org/10.1007/978-1-4939-8730-6_18
141. Mauro VP (2018) Codon optimization in the production of recombinant biotherapeutics: potential risks and considerations. *BioDrugs* 32(1):69–81. <https://doi.org/10.1007/s40259-018-0261-x>
142. Mandad S, Rahman RU, Centeno TP, Vidal RO, Wildhagen H, Rammner B, Keihani S, Opazo F, Urban I, Ischebeck T, Kirli K, Benito E, Fischer A, Yousefi RY, Dennerlein S, Rehling P, Feussner I, Urlaub H, Bonn S, Rizzoli SO, Fornasiero EF (2018) The codon sequences predict protein lifetimes and other parameters of the protein life cycle in the mouse brain. *Sci Rep* 8(1):16913. <https://doi.org/10.1038/s41598-018-35277-8>
143. Liu Y, Mi Y, Mueller T, Kreibich S, Williams EG, Van Drogen A, Borel C, Frank M, Germain PL, Bludau I, Mehnert M, Seifert M, Emmenlauer M, Sorg I, Bezrukov F, Bena FS, Zhou H, Dehio C, Testa G, Saez-Rodriguez J, Antonarakis SE, Hardt WD, Aebersold R (2019) Multi-omic measurements of heterogeneity in HeLa cells across laboratories. *Nat Biotechnol* 37(3):314–322. <https://doi.org/10.1038/s41587-019-0037-y>

144. Xu Y, Ma P, Shah P, Rokas A, Liu Y, Johnson CH (2013) Non-optimal codon usage is a mechanism to achieve circadian clock conditionality. *Nature* 495(7439):116–120. <https://doi.org/10.1038/nature11942>
145. Yourono J, Tanemura S (1970) Restoration of in-phase translation by an unlinked suppressor of a frameshift mutation in *Salmonella typhimurium*. *Nature* 225(5231):422–426
146. Bossi L, Roth JR (1980) The influence of codon context on genetic code translation. *Nature* 286(5769):123–127
147. Giliberti J, O'Donnell S, Etten WJ, Janssen GR (2012) A 5'-terminal phosphate is required for stable ternary complex formation and translation of leaderless mRNA in *Escherichia coli*. *RNA* 18(3):508–518. <https://doi.org/10.1261/rna.027698.111>
148. Akulich KA, Andreev DE, Terenin IM, Smirnova VV, Anisimova AS, Makeeva DS, Arkhipova VI, Stolboushkina EA, Garber MB, Prokofjeva MM, Spirin PV, Prassolov VS, Shatsky IN, Dmitriev SE (2016) Four translation initiation pathways employed by the leaderless mRNA in eukaryotes. *Sci Rep* 28(6):37905. <https://doi.org/10.1038/srep37905>
149. Brar GA (2016) Beyond the triplet code: context cues transform translation. *Cell* 167(7):1681–1692. <https://doi.org/10.1016/j.cell.2016.09.022>
150. Li GW, Oh E, Weissman JS (2012) The anti-Shine-Dalgarno sequence drives translational pausing and codon choice in bacteria. *Nature* 484(7395):538–541. <https://doi.org/10.1038/nature10965>
151. Yurovsky A, Amin MR, Gardin J, Chen Y, Skiena S, Futcher B (2018) Prokaryotic coding regions have little if any specific depletion of Shine-Dalgarno motifs. *PLoS ONE* 13(8):e0202768. <https://doi.org/10.1371/journal.pone.0202768>
152. Mohammad F, Woolstenhulme CJ, Green R, Buskirk AR (2016) Clarifying the translational pausing landscape in bacteria by ribosome profiling. *Cell Rep* 14(4):686–694. <https://doi.org/10.1016/j.celrep.2015.12.073>
153. Mohammad F, Green R, Buskirk AR (2019) A systematically-revised ribosome profiling method for bacteria reveals pauses at single-codon resolution. *Elife* 8:pii: e42591. <https://doi.org/10.7554/elife.42591>
154. Ogle JM, Ramakrishnan V (2005) Structural insights into translational fidelity. *Annu Rev Biochem* 74:129–177
155. Gamper HB, Masuda I, Frenkel-Morgenstern M, Hou YM (2015) The UGG isoacceptor of tRNAPro is naturally prone to frameshifts. *Int J Mol Sci* 16(7):14866–14883. <https://doi.org/10.3390/ijms160714866>
156. Gamper HB, Masuda I, Frenkel-Morgenstern M, Hou YM (2015) Maintenance of protein synthesis reading frame by EF-P and m(1)G37-tRNA. *Nat Commun* 26(6):7226. <https://doi.org/10.1038/ncomms8226>
157. Gutman GA, Hatfield GW (1989) Nonrandom utilization of codon pairs in *Escherichia coli*. *Proc Natl Acad Sci U S A* 86(10):3699–3703
158. Fedorov A, Saxonov S, Gilbert W (2002) Regularities of context-dependent codon bias in eukaryotic genes. *Nucleic Acids Res* 30(5):1192–1197
159. Ciandrini L, Stansfield I, Romano MC (2013) Ribosome traffic on mRNAs maps to gene ontology: genome-wide quantification of translation initiation rates and polysome size regulation. *PLoS Comput Biol* 9(1):e1002866. <https://doi.org/10.1371/journal.pcbi.1002866>
160. Letzring DP, Wolf AS, Brule CE, Grayhack EJ (2013) Translation of CGA codon repeats in yeast involves quality control components and ribosomal protein L1. *RNA* 19(9):1208–1217. <https://doi.org/10.1261/rna.039446.113>
161. Chevance FF, Le Guyon S, Hughes KT (2014) The effects of codon context on in vivo translation speed. *PLoS Genet* 10(6):e1004392. <https://doi.org/10.1371/journal.pgen.1004392>
162. Coleman JR, Papamichail D, Skiena S, Futcher B, Wimmer E, Mueller S (2008) Virus attenuation by genome-scale changes in codon pair bias. *Science* 320(5884):1784–1787. <https://doi.org/10.1126/science.1155761>
163. Tulloch F, Atkinson NJ, Evans DJ, Ryan MD, Simmonds P (2014) RNA virus attenuation by codon pair deoptimisation is an artefact of increases in CpG/UpA dinucleotide frequencies. *Elife* 9(3):e04531. <https://doi.org/10.7554/eLife.04531>

164. Peil L, Starosta AL, Lassak J, Atkinson GC, Virumäe K, Spitzer M, Tenson T, Jung K, Remme J, Wilson DN (2013) Distinct XPPX sequence motifs induce ribosome stalling, which is rescued by the translation elongation factor EF-P. *Proc Natl Acad Sci U S A.* 110(38):15265–15270. <https://doi.org/10.1073/pnas.1310642110>
165. Starosta AL, Lassak J, Peil L, Atkinson GC, Virumäe K, Tenson T, Remme J, Jung K, Wilson DN (2014) Translational stalling at polyproline stretches is modulated by the sequence context upstream of the stall site. *Nucleic Acids Res* 42(16):10711–10719. <https://doi.org/10.1093/nar/gku768>
166. Gamble CE, Brule CE, Dean KM, Fields S, Grayhack EJ (2016) Adjacent codons act in concert to modulate translation efficiency in yeast. *Cell* 166(3):679–690. <https://doi.org/10.1016/j.cell.2016.05.070>
167. McCarthy C, Carrea A, Diambra L (2017) Bicodon bias can determine the role of synonymous SNPs in human diseases. *BMC Genom* 18(1):227. <https://doi.org/10.1186/s12864-017-3609-6>
168. Chevance FFV, Hughes KT (2017) Case for the genetic code as a triplet of triplets. *Proc Natl Acad Sci U S A.* 114(18):4745–4750. <https://doi.org/10.1073/pnas.1614896114>
169. Ghoneim DH, Zhang X, Brule CE, Mathews DH, Grayhack EJ (2018) Conservation of location of several specific inhibitory codon pairs in the *Saccharomyces sensu stricto* yeasts reveals translational selection. *Nucleic Acids Res.* <https://doi.org/10.1093/nar/gky1262>
170. Komar AA, Lesnik T, Reiss C (1999) Synonymous codon substitutions affect ribosome traffic and protein folding during in vitro translation. *FEBS Lett* 462(3):387–391
171. Zhang G, Hubalewska M, Ignatova Z (2009) Transient ribosomal attenuation coordinates protein synthesis and co-translational folding. *Nat Struct Mol Biol* 16(3):274–280. <https://doi.org/10.1038/nsmb.1554>
172. Buhr F, Jha S, Thommen M, Mittelstaet J, Kutz F, Schwalbe H, Rodnina MV, Komar AA (2016) Synonymous codons direct cotranslational folding toward different protein conformations. *Mol Cell* 61(3):341–351. <https://doi.org/10.1016/j.molcel.2016.01.008>
173. Pechmann S, Chartron JW, Frydman J (2014) Local slowdown of translation by nonoptimal codons promotes nascent-chain recognition by SRP in vivo. *Nat Struct Mol Biol* 21(12):1100–1105. <https://doi.org/10.1038/nsmb.2919>
174. Lee Y, Zhou T, Tartaglia GG, Vendruscolo M, Wilke CO (2010) Translationally optimal codons associate with aggregation-prone sites in proteins. *Proteomics* 10(23):4163–4171. <https://doi.org/10.1002/pmic.201000229>
175. Cannarozzi G, Schraudolph NN, Faty M, von Rohr P, Friberg MT, Roth AC, Gonnet P, Gonnet G, Barral Y (2010) A role for codon order in translation dynamics. *Cell* 141(2):355–367. <https://doi.org/10.1016/j.cell.2010.02.036>
176. Carrier MJ, Buckingham RH (1984) An effect of codon context on the mistranslation of UGU codons in vitro. *J Mol Biol* 175(1):29–38
177. Buckingham RH (1994) Codon context and protein synthesis: enhancements of the genetic code. *Biochimie* 76(5):351–354
178. Baranov PV, Atkins JF, Yordanova MM (2015) Augmented genetic decoding: global, local and temporal alterations of decoding processes and codon meaning. *Nat Rev Genet* 16(9):517–529. <https://doi.org/10.1038/nrg3963>
179. Skuzeski JM, Nichols LM, Gesteland RF, Atkins JF (1991) The signal for a leaky UAG stop codon in several plant viruses includes the two downstream codons. *J Mol Biol* 218(2):365–373
180. Chan CS, Jungreis I, Kellis M (2013) Heterologous stop codon readthrough of metazoan readthrough candidates in yeast. *PLoS ONE* 8(3):e59450. <https://doi.org/10.1371/journal.pone.0059450>
181. Loughran G, Jungreis I, Tzani I, Power M, Dmitriev RI, Ivanov IP, Kellis M, Atkins JF (2018) Stop codon readthrough generates a C-terminally extended variant of the human vitamin D receptor with reduced calcitriol response. *J Biol Chem* 293(12):4434–4444. <https://doi.org/10.1074/jbc.M117.818526>

182. Jungreis I, Lin MF, Spokony R, Chan CS, Negre N, Victorsen A, White KP, Kellis M (2011) Evidence of abundant stop codon readthrough in *Drosophila* and other metazoa. *Genome Res* 21(12):2096–2113. <https://doi.org/10.1101/gr.119974.110>
183. Jungreis I, Chan CS, Waterhouse RM, Fields G, Lin MF, Kellis M (2016) Evolutionary dynamics of abundant stop codon readthrough. *Mol Biol Evol* 33(12):3108–3132
184. Rajput B, Pruitt KD, Murphy TD (2019) RefSeq curation and annotation of stop codon recoding in vertebrates. *Nucleic Acids Res* 47(2):594–606. <https://doi.org/10.1093/nar/gky1234>
185. Swart EC, Serra V, Petroni G, Nowacki M (2016) Genetic codes with no dedicated stop codon: context-dependent translation termination. *Cell* 166(3):691–702. <https://doi.org/10.1016/j.cell.2016.06.020>
186. Belew AT, Dinman JD (2015) Cell cycle control (and more) by programmed –1 ribosomal frameshifting: implications for disease and therapeutics. *Cell Cycle* 14(2):172–178. <https://doi.org/10.4161/15384101.2014.989123>
187. Belew AT, Hepler NL, Jacobs JL, Dinman JD (2008) PRFdb: a database of computationally predicted eukaryotic programmed –1 ribosomal frameshift signals. *BMC Genom* 17(9):339. <https://doi.org/10.1186/1471-2164-9-339>
188. Pinheiro M, Afreixo V, Moura G, Freitas A, Santos MA, Oliveira JL (2006) Statistical, computational and visualization methodologies to unveil gene primary structure features. *Methods Inf Med* 45(2):163–168
189. Moura G, Pinheiro M, Silva R, Miranda I, Afreixo V, Dias G, Freitas A, Oliveira JL, Santos MA (2005) Comparative context analysis of codon pairs on an ORFeome scale. *Genome Biol* 6(3):R28
190. Moura G, Pinheiro M, Arrais J, Gomes AC, Carreto L, Freitas A, Oliveira JL, Santos MA (2007) Large scale comparative codon-pair context analysis unveils general rules that fine-tune evolution of mRNA primary structure. *PLoS ONE* 2(9):e847
191. Tats A, Tenson T, Remm M (2008) Preferred and avoided codon pairs in three domains of life. *BMC Genom* 8(9):463. <https://doi.org/10.1186/1471-2164-9-463>
192. Doyle F, Leonardi A, Endres L, Tenenbaum SA, Dedon PC, Begley TJ (2016) Gene- and genome-based analysis of significant codon patterns in yeast, rat and mice genomes with the CUT Codon UTilization tool. *Methods* 1(107):98–109. <https://doi.org/10.1016/jymeth.2016.05.010>
193. Alexaki A, Kames J, Holcomb DD, Athey J, Santana-Quintero LV, Lam PVN, Hamasaki-Katagiri N, Osipova E, Simonyan V, Bar H, Komar AA, Kimchi-Sarfaty C (2019) Codon and codon-pair usage tables (CoCoPUTs): facilitating genetic variation analyses and recombinant gene design. *J Mol Biol* pii: S0022-2836(19)30228-1. <https://doi.org/10.1016/j.jmb.2019.04.021>
194. Kucukyildirim S, Long H, Sung W, Miller SF, Doak TG, Lynch M (2016) The rate and spectrum of spontaneous mutations in *Mycobacterium smegmatis*, a bacterium naturally devoid of the postreplicative mismatch repair pathway. *G3 (Bethesda)* 6(7):2157–2163. <https://doi.org/10.1534/g3.116.030130>
195. Aslam S, Lan XR, Zhang BW, Chen ZL, Wang L, Niu DK (2019) Aerobic prokaryotes do not have higher GC contents than anaerobic prokaryotes, but obligate aerobic prokaryotes have. *BMC Evol Biol* 19(1):35. <https://doi.org/10.1186/s12862-019-1365-8>
196. Hershberg R, Petrov DA (2010) Evidence that mutation is universally biased towards AT in bacteria. *PLoS Genet* 6(9):e1001115. <https://doi.org/10.1371/journal.pgen.1001115>
197. Lassalle F, Périan S, Bataillon T, Nesme X, Duret L, Daubin V (2015) GC-Content evolution in bacterial genomes: the biased gene conversion hypothesis expands. *PLoS Genet* 11(2):e1004941. <https://doi.org/10.1371/journal.pgen.1004941>
198. Hildebrand F, Meyer A, Eyre-Walker A (2010) Evidence of selection upon genomic GC-content in bacteria. *PLoS Genet* 6(9):e1001107. <https://doi.org/10.1371/journal.pgen.1001107>
199. Bobay LM, Ochman H (2017) Impact of recombination on the base composition of bacteria and archaea. *Mol Biol Evol* 34(10):2627–2636. <https://doi.org/10.1093/molbev/msx189>

200. Trotta E (2016) Selective forces and mutational biases drive stop codon usage in the human genome: a comparison with sense codon usage. *BMC Genom* 17(17):366. <https://doi.org/10.1186/s12864-016-2692-4>
201. Wilke CO, Drummond DA (2006) Population genetics of translational robustness. *Genetics* 173(1):473–481
202. Zhou T, Weems M, Wilke CO (2009) Translationally optimal codons associate with structurally sensitive sites in proteins. *Mol Biol Evol* 26(7):1571–1580. <https://doi.org/10.1093/molbev/msp070>
203. Yu CH, Dang Y, Zhou Z, Wu C, Zhao F, Sachs MS, Liu Y (2015) Codon usage influences the local rate of translation elongation to regulate co-translational protein folding. *Mol Cell* 59(5):744–754. <https://doi.org/10.1016/j.molcel.2015.07.018>
204. Yan X, Hoek TA, Vale RD, Tanenbaum ME (2016) Dynamics of translation of single mRNA molecules in vivo. *Cell* 165(4):976–989. <https://doi.org/10.1016/j.cell.2016.04.034>
205. Zhao F, Yu CH, Liu Y (2017) Codon usage regulates protein structure and function by affecting translation elongation speed in *Drosophila* cells. *Nucleic Acids Res* 45(14):8484–8492. <https://doi.org/10.1093/nar/gkx501>
206. Li GW, Burkhardt D, Gross C, Weissman JS (2014) Quantifying absolute protein synthesis rates reveals principles underlying allocation of cellular resources. *Cell* 157(3):624–635. <https://doi.org/10.1016/j.cell.2014.02.033>
207. Shah P, Gilchrist MA (2010) Effect of correlated tRNA abundances on translation errors and evolution of codon usage bias. *PLoS Genet* 6(9):e1001128. <https://doi.org/10.1371/journal.pgen.1001128>
208. Chamary JV, Parmley JL, Hurst LD (2006) Hearing silence: non-neutral evolution at synonymous sites in mammals. *Nat Rev Genet* 7(2):98–108
209. Sauna ZE, Kimchi-Sarfaty C (2011) Understanding the contribution of synonymous mutations to human disease. *Nat Rev Genet* 12(10):683–691. <https://doi.org/10.1038/nrg3051>
210. Kirchner S, Cai Z, Rauscher R, Kastelic N, Anding M, Czech A, Kleizen B, Ostedgaard LS, Braakman I, Sheppard DN, Ignatova Z (2017) Alteration of protein function by a silent polymorphism linked to tRNA abundance. *PLoS Biol* 15(5):e2000779. <https://doi.org/10.1371/journal.pbio.2000779>
211. Zhou Z, Dang Y, Zhou M, Li L, Yu CH, Fu J, Chen S, Liu Y (2016) Codon usage is an important determinant of gene expression levels largely through its effects on transcription. *Proc Natl Acad Sci U S A* 113(41):E6117–E6125
212. Mittal P, Brindle J, Stephen J, Plotkin JB, Kudla G (2018) Codon usage influences fitness through RNA toxicity. *Proc Natl Acad Sci U S A* 115(34):8639–8644. <https://doi.org/10.1073/pnas.1810022115>
213. Weinberg DE, Shah P, Eichhorn SW, Hussmann JA, Plotkin JB, Bartel DP (2016) Improved ribosome-footprint and mRNA measurements provide insights into dynamics and regulation of yeast translation. *Cell Rep* 14(7):1787–1799. <https://doi.org/10.1016/j.celrep.2016.01.043>
214. Chu D, Kazana E, Bellanger N, Singh T, Tuite MF, von der Haar T (2014) Translation elongation can control translation initiation on eukaryotic mRNAs. *EMBO J* 33(1):21–34. <https://doi.org/10.1002/embj.201385651>
215. Chan LY, Mugler CF, Heinrich S, Vallotton P, Weis K (2018) Non-invasive measurement of mRNA decay reveals translation initiation as the major determinant of mRNA stability. *Elife* 7:pii: e32536. <https://doi.org/10.7554/elife.32536>
216. Eraslan B, Wang D, Gusic M, Prokisch H, Hallström BM, Uhlén M, Asplund A, Pontén F, Wieland T, Hopf T, Hahne H, Kuster B, Gagneur J (2019) Quantification and discovery of sequence determinants of protein-per-mRNA amount in 29 human tissues. *Mol Syst Biol* 15(2):e8513. <https://doi.org/10.15252/msb.20188513>
217. Zhou M, Guo J, Cha J, Chae M, Chen S, Barral JM, Sachs MS, Liu Y (2013) Non-optimal codon usage affects expression, structure and function of clock protein FRQ. *Nature* 495(7439):111–115. <https://doi.org/10.1038/nature11833>
218. Chan C, Pham P, Dedon PC, Begley TJ (2018) Lifestyle modifications: coordinating the tRNA epitranscriptome with codon bias to adapt translation during stress responses. *Genome Biol* 19(1):228. <https://doi.org/10.1186/s13059-018-1611-1>

219. Novoa EM, Pavon-Eternod M, Pan T, de Pouplana LR (2012) A role for tRNA modifications in genome structure and codon usage. *Cell* 149(1):202–213. <https://doi.org/10.1016/j.cell.2012.01.050>
220. Fuglsang A (2005) Intron position of UUA codons in streptomycetes. *Microbiology* 151(Pt 10):3150–3152
221. Zaburanny N, Ostash B, Fedorenko V (2009) TTA Lynx: a web-based service for analysis of actinomycete genes containing rare TTA codon. *Bioinformatics* 25(18):2432–2433. <https://doi.org/10.1093/bioinformatics/btp402>
222. Jee J, Rasouly A, Shamovsky I, Akivis Y, Steinman SR, Mishra B, Nudler E (2016) Rates and mechanisms of bacterial mutagenesis from maximum-depth sequencing. *Nature* 534(7609):693–696
223. Kosiol C, Goldman N (2011) Markovian and non-Markovian protein sequence evolution: aggregated Markov process models. *J Mol Biol* 411(4):910–923. <https://doi.org/10.1016/j.jmb.2011.06.005>
224. Anisimova M, Kosiol C (2009) Investigating protein-coding sequence evolution with probabilistic codon substitution models. *Mol Biol Evol* 26(2):255–271. <https://doi.org/10.1093/molbev/msn232>
225. Schneider A, Cannarozzi GM, Gonnet GH (2005) Empirical codon substitution matrix. *BMC Bioinform* 1(6):134
226. Beaumont MA, Rannala B (2004) The Bayesian revolution in genetics. *Nat Rev Genet* 5(4):251–261
227. Eddy SR (2004) What is Bayesian statistics? *Nat Biotechnol* 22(9):1177–1178
228. Do CB, Batzoglou S (2008) What is the expectation maximization algorithm? *Nat Biotechnol* 26(8):897–899. <https://doi.org/10.1038/nbt1406>
229. Anisimova M, Bielawski JP, Yang Z (2001) Accuracy and power of the likelihood ratio test in detecting adaptive molecular evolution. *Mol Biol Evol* 18(8):1585–1592
230. Akaike H (1974) A new look at the statistical model identification. *IEEE Trans Autom Control* 19(6):716–723. <https://doi.org/10.1109/TAC.1974.1100705>
231. Davydov II, Salamin N, Robinson-Rechavi M (2019) Large-scale comparative analysis of codon models accounting for protein and nucleotide selection. *Mol Biol Evol* pii: msz048. <https://doi.org/10.1093/molbev/msz048>
232. Arenas M (2015) Trends in substitution models of molecular evolution. *Front Genet* 26(6):319. <https://doi.org/10.3389/fgene.2015.00319>
233. Yang Z (2006) Computational molecular evolution. Oxford University Press, Oxford, 324 p. ISBN 978–0–19–856699–1
234. Venkat A, Hahn MW, Thornton JW (2018) Multinucleotide mutations cause false inferences of lineage-specific positive selection. *Nat Ecol Evol* 2(8):1280–1288. <https://doi.org/10.1038/s41559-018-0584-5>
235. Liu X, Liu H, Guo W, Yu K (2012) Codon substitution models based on residue similarity and their applications. *Gene* 509(1):136–141. <https://doi.org/10.1016/j.gene.2012.07.075>
236. Delport W, Scheffler K, Botha G, Gravenor MB, Muse SV, Kosakovsky Pond SL (2010) CodonTest: modeling amino acid substitution preferences in coding sequences. *PLoS Comput Biol* 6(8):pii: e1000885. <https://doi.org/10.1371/journal.pcbi.1000885>
237. Huttley GA (2004) Modeling the impact of DNA methylation on the evolution of BRCA1 in mammals. *Mol Biol Evol* 21(9):1760–1768
238. Mayrose I, Doron-Faigenboim A, Bacharach E, Pupko T (2007) Towards realistic codon models: among site variability and dependency of synonymous and non-synonymous rates. *Bioinformatics* 23(13):i319–i327
239. Higgs PG, Hao W, Golding GB (2007) Identification of conflicting selective effects on highly expressed genes. *Evol Bioinform Online* 14(3):1–13
240. Kubatko L, Shah P, Herbei R, Gilchrist MA (2016) A codon model of nucleotide substitution with selection on synonymous codon usage. *Mol Phylogenet Evol* 94(Pt A):290–297. <https://doi.org/10.1016/j.ympev.2015.08.026>

241. Beaulieu JM, O'Meara BC, Zaretzki R, Landerer C, Chai J, Gilchrist MA (2019) Population genetics based phylogenetics under stabilizing selection for an optimal amino acid sequence: a nested modeling approach. *Mol Biol Evol* 36(4):834–851. <https://doi.org/10.1093/molbev/msy222>
242. Higgs PG (2008) Linking population genetics to phylogenetics. *Banach Center Publ* 80(1):145–166
243. Pouyet F, Bailly-Béchet M, Mouchiroud D, Guéguen L (2016) SENCA: a multilayered codon model to study the origins and dynamics of codon usage. *Genome Biol Evol* 8(8):2427–2441. <https://doi.org/10.1093/gbe/evw165>
244. Rodrigue N, Lartillot N (2017) Detecting adaptation in protein-coding genes using a bayesian site-heterogeneous mutation-selection codon substitution model. *Mol Biol Evol* 34(1):204–214. <https://doi.org/10.1093/molbev/msw220>
245. Teufel AI, Ritchie AM, Wilke CO, Liberles DA (2018) Using the mutation-selection framework to characterize selection on protein sequences. *Genes (Basel)* 9(8):pii: E409. <https://doi.org/10.3390/genes9080409>
246. Dunn KA, Kenney T, Gu H, Bielawski JP (2019) Improved inference of site-specific positive selection under a generalized parametric codon model when there are multinucleotide mutations and multiple nonsynonymous rates. *BMC Evol Biol* 19(1):22. <https://doi.org/10.1186/s12862-018-1326-7>
247. Dayhoff MO, Schwartz RM, Orcutt BC (1978) A model of evolutionary change in proteins. In: *Atlas of protein sequence and structure*, vol 5, pp 345–352
248. Gonnet GH, Cohen MA, Benner SA (1992) Exhaustive matching of the entire protein sequence database. *Science* 256(5062):1443–1445
249. De Maio N, Holmes I, Schlötterer C, Kosiol C (2013) Estimating empirical codon hidden Markov models. *Mol Biol Evol* 30(3):725–736. <https://doi.org/10.1093/molbev/mss266>
250. Zoller S, Schneider A (2010) Empirical analysis of the most relevant parameters of codon substitution models. *J Mol Evol* 70(6):605–612. <https://doi.org/10.1007/s00239-010-9356-9>
251. Kosiol C, Holmes I, Goldman N (2007) An empirical codon model for protein sequence evolution. *Mol Biol Evol* 24(7):1464–1479
252. Doron-Faigenboim A, Pupko T (2007) A combined empirical and mechanistic codon model. *Mol Biol Evol* 24(2):388–397
253. Zoller S, Schneider A (2012) A new semiempirical codon substitution model based on principal component analysis of mammalian sequences. *Mol Biol Evol* 29(1):271–277. <https://doi.org/10.1093/molbev/msr198>
254. Hoban S, Bertorelle G, Gaggiotti OE (2012) Computer simulations: tools for population and evolutionary genetics. *Nat Rev Genet* 13(2):110–122. <https://doi.org/10.1038/nrg3130>
255. Arenas M (2013) Computer programs and methodologies for the simulation of DNA sequence data with recombination. *Front Genet* 14(9). <https://doi.org/10.3389/fgene.2013.00009>
256. Anisimova M, Nielsen R, Yang Z (2003) Effect of recombination on the accuracy of the likelihood method for detecting positive selection at amino acid sites. *Genetics* 164(3):1229–1236
257. Dalquen DA, Anisimova M, Gonnet GH, Dessimoz C (2012) ALF—a simulation framework for genome evolution. *Mol Biol Evol* 29(4):1115–1123. <https://doi.org/10.1093/molbev/msr268>
258. Arenas M, Posada D (2014) Simulation of genome-wide evolution under heterogeneous substitution models and complex multispecies coalescent histories. *Mol Biol Evol* 31(5):1295–1301. <https://doi.org/10.1093/molbev/msu078>
259. Mallo D, De Oliveira Martins L, Posada D (2016) SimPhy: phylogenomic simulation of gene, locus, and species trees. *Syst Biol* 65(2):334–344. <https://doi.org/10.1093/sysbio/syv082>
260. Haller BC, Messer PW (2019) SLiM 3: forward genetic simulations beyond the Wright-Fisher model. *Mol Biol Evol* 36(3):632–637. <https://doi.org/10.1093/molbev/msy228>
261. Klosterman PS, Uzilov AV, Bendaña YR, Bradley RK, Chao S, Kosiol C, Goldman N, Holmes I (2006) XRate: a fast prototyping, training and annotation tool for phylo-grammars. *BMC Bioinform* 3(7):428

262. Barquist L, Holmes I (2008) xREI: a phylo-grammar visualization webserver. Nucleic Acids Res 36(Web Server issue):W65–W69. <https://doi.org/10.1093/nar/gkn283>
263. Wernersson R, Pedersen AG (2003) RevTrans: multiple alignment of coding DNA from aligned amino acid sequences. Nucleic Acids Res 31(13):3537–3539
264. Ranwez V, Douzery EJP, Cambon C, Chantret N, Delsuc F (2018) MACSE v2: toolkit for the alignment of coding sequences accounting for frameshifts and stop codons. Mol Biol Evol 35(10):2582–2584. <https://doi.org/10.1093/molbev/msy159>
265. Noens EE, Mersinias V, Traag BA, Smith CP, Koerten HK, van Wezel GP (2005) SsgA-like proteins determine the fate of peptidoglycan during sporulation of *Streptomyces coelicolor*. Mol Microbiol 58(4):929–944
266. Rabyk M, Yushchuk O, Rokytkskyy I, Anisimova M, Ostash B (2018) Genomic insights into evolution of AdpA family master regulators of morphological differentiation and secondary metabolism in *Streptomyces*. J Mol Evol 86(3–4):204–215. <https://doi.org/10.1007/s00239-018-9834-z>
267. Wang M, Kapralov MV, Anisimova M (2011) Coevolution of amino acid residues in the key photosynthetic enzyme Rubisco. BMC Evol Biol 23(11):266. <https://doi.org/10.1186/1471-2148-11-266>
268. Kapralov MV, Filatov DA (2007) Widespread positive selection in the photosynthetic Rubisco enzyme. BMC Evol Biol 11(7):73
269. Elena SF, Lenski RE (2003) Evolution experiments with microorganisms: the dynamics and genetic bases of adaptation. Nat Rev Genet 4(6):457–469
270. Charlesworth B (2013) Stabilizing selection, purifying selection, and mutational bias in finite populations. Genetics 194(4):955–971. <https://doi.org/10.1534/genetics.113.151555>
271. Kimura M (1991) Recent development of the neutral theory viewed from the Wrightian tradition of theoretical population genetics. Proc Natl Acad Sci U S A 88(14):5969–5973
272. Jensen JD, Payseur BA, Stephan W, Aquadro CF, Lynch M, Charlesworth D, Charlesworth B (2019) The importance of the Neutral Theory in 1968 and 50 years on: a response to Kern and Hahn 2018. Evolution 73(1):111–114. <https://doi.org/10.1111/evo.13650>
273. Kimura M (1981) Possibility of extensive neutral evolution under stabilizing selection with special reference to nonrandom usage of synonymous codons. Proc Natl Acad Sci U S A 78(9):5773–5777
274. Fuller ZL, Haynes GD, Zhu D, Batterton M, Chao H, Dugan S, Javaid M, Jayaseelan JC, Lee S, Li M, Ongeri F, Qi S, Han Y, Doddapaneni H, Richards S, Schaeffer SW (2014) Evidence for stabilizing selection on codon usage in chromosomal rearrangements of *Drosophila pseudoobscura*. G3 (Bethesda) 4(12):2433–2449. <https://doi.org/10.1534/g3.114.014860>
275. Jackson BC, Campos JL, Haddrill PR, Charlesworth B, Zeng K (2017) Variation in the intensity of selection on codon bias over time causes contrasting patterns of base composition evolution in *Drosophila*. Genome Biol Evol 9(1):102–123. <https://doi.org/10.1093/gbe/evw291>
276. Plotkin JB, Dushoff J, Fraser HB (2004) Detecting selection using a single genome sequence of *M. tuberculosis* and *P. falciparum*. Nature 428(6986):942–945
277. Plotkin JB, Dushoff J, Desai MM, Fraser HB (2006) Codon usage and selection on proteins. J Mol Evol 63(5):635–653
278. Zhang J (2005) On the evolution of codon volatility. Genetics 169(1):495–501
279. Dagan T, Graur D (2005) The comparative method rules! Codon volatility cannot detect positive Darwinian selection using a single genome sequence. Mol Biol Evol 22(3):496–500
280. O'Connell MJ, Doyle AM, Juenger TE, Donoghue MT, Keshavaiah C, Tuteja R, Spillane C (2012) In *Arabidopsis thaliana* codon volatility scores reflect GC3 composition rather than selective pressure. BMC Res Notes 17(5):359. <https://doi.org/10.1186/1756-0500-5-359>
281. Tajima F (1989) Statistical method for testing the neutral mutation hypothesis by DNA polymorphism. Genetics 123(3):585–595
282. McDonald JH, Kreitman M (1991) Adaptive protein evolution at the Adh locus in *Drosophila*. Nature 351(6328):652–654

283. Zhai W, Slatkin M, Nielsen R (2007) Exploring variation in the d(N)/d(S) ratio among sites and lineages using mutational mappings: applications to the influenza virus. *J Mol Evol* 65(3):340–348
284. Gelman A, Meng X-L, Stern H (1996) Posterior predictive assessment of model fitness via realized discrepancies. *Stat Sin* 6:733–807
285. Kosakovsky Pond SL, Frost SD (2005) Not so different after all: a comparison of methods for detecting amino acid sites under selection. *Mol Biol Evol* 22(5):1208–1222
286. Lemey P, Minin VN, Bielejec F, Kosakovsky Pond SL, Suchard MA (2012) A counting renaissance: combining stochastic mapping and empirical Bayes to quickly detect amino acid sites under positive selection. *Bioinformatics* 28(24):3248–3256. <https://doi.org/10.1093/bioinformatics/bts580>
287. Yang Z, Nielsen R (2000) Estimating synonymous and nonsynonymous substitution rates under realistic evolutionary models. *Mol Biol Evol* 17(1):32–43
288. Gil M, Zanetti MS, Zoller S, Anisimova M (2013) CodonPhyML: fast maximum likelihood phylogeny estimation under codon substitution models. *Mol Biol Evol* 30(6):1270–1280. <https://doi.org/10.1093/molbev/mst034>
289. Hedge J, Wilson DJ (2016) Practical approaches for detecting selection in microbial genomes. *PLoS Comput Biol* 12(2):e1004739. <https://doi.org/10.1371/journal.pcbi.1004739>
290. Yang Z (2007) PAML 4: phylogenetic analysis by maximum likelihood. *Mol Biol Evol* 24(8):1586–1591
291. Gao F, Chen C, Arab DA, Du Z, He Y, Ho SYW (2019) EasyCodeML: a visual tool for analysis of selection using CodeML. *Ecol Evol* 9(7):3891–3898. <https://doi.org/10.1002/ece3.5015>
292. Zhao K, Henderson E, Bullard K, Oberste MS, Burns CC, Jorba J (2018) PoSE: visualization of patterns of sequence evolution using PAML and MATLAB. *BMC Bioinform* 19(Suppl 11):364. <https://doi.org/10.1186/s12859-018-2335-7>
293. Pond SL, Frost SD, Muse SV (2005) HyPhy: hypothesis testing using phylogenies. *Bioinformatics* 21(5):676–679
294. Weaver S, Shank SD, Spielman SJ, Li M, Muse SV, Kosakovsky Pond SL (2018) Datamonkey 2.0: a modern web application for characterizing selective and other evolutionary processes. *Mol Biol Evol* 35:773–777. <https://doi.org/10.1093/molbev/msx335>
295. Bouckaert R, Vaughan TG, Barido-Sottani J, Duchêne S, Fourment M, Gavryushkina A, Heled J, Jones G, Kühnert D, De Maio N, Matschiner M, Mendes FK, Müller NF, Ogilvie HA, du Plessis L, Popinga A, Rambaut A, Rasmussen D, Siveroni I, Suchard MA, Wu CH, Xie D, Zhang C, Stadler T, Drummond AJ (2019) BEAST 2.5: an advanced software platform for Bayesian evolutionary analysis. *PLoS Comput Biol* 15(4):e1006650. <https://doi.org/10.1371/journal.pcbi.1006650>
296. Sealfon RS, Lin MF, Jungreis I, Wolf MY, Kellis M, Sabeti PC (2015) FRESCO: finding regions of excess synonymous constraint in diverse viruses. *Genome Biol* 17(16):38. <https://doi.org/10.1186/s13059-015-0603-7>
297. Stern A, Doron-Faigenboim A, Erez E, Martz E, Bacharach E, Pupko T (2007) Selecton 2007: advanced models for detecting positive and purifying selection using a Bayesian inference approach. *Nucleic Acids Res* 35(Web Server issue):W506–W511
298. Supék F, Šmuc T (2010) On relevance of codon usage to expression of synthetic and natural genes in *Escherichia coli*. *Genetics* 185(3):1129–1134. <https://doi.org/10.1534/genetics.110.115477>
299. Pokusaeva VO, Usanova DR, Putintseva EV, Espinar L, Sarkisyan KS, Mishin AS, Bogatyreva NS, Ivankov DN, Akopyan AV, Avvakumov SY, Povolotskaya IS, Filion GJ, Carey LB, Kondrashov FA (2019) An experimental assay of the interactions of amino acids from orthologous sequences shaping a complex fitness landscape. *PLoS Genet* 15(4):e1008079. <https://doi.org/10.1371/journal.pgen.1008079>
300. Darriba D, Flouri T, Stamatakis A (2018) The state of software for evolutionary biology. *Mol Biol Evol* 35(5):1037–1046. <https://doi.org/10.1093/molbev/msy014>
301. Abadi S, Azouri D, Pupko T, Mayrose I (2019) Model selection may not be a mandatory step for phylogeny reconstruction. *Nat Commun* 10(1):934. <https://doi.org/10.1038/s41467-019-08822-w>

302. Spielman SJ, Kosakovsky Pond SL (2018) Relative evolutionary rates in proteins are largely insensitive to the substitution model. *Mol Biol Evol*. <https://doi.org/10.1093/molbev/msy127>
303. Chionh YH, McBee M, Babu IR, Hia F, Lin W, Zhao W, Cao J, Dziergowska A, Malkiewicz A, Begley TJ, Alonso S, Dedon PC (2016) tRNA-mediated codon-biased translation in mycobacterial hypoxic persistence. *Nat Commun* 11(7):13302. <https://doi.org/10.1038/ncomms13302>
304. Gingold H, Tehler D, Christoffersen NR, Nielsen MM, Asmar F, Kooistra SM, Christoffersen NS, Christensen LL, Borre M, Sørensen KD, Andersen LD, Andersen CL, Hulleman E, Wurdinger T, Ralfkiaer E, Helin K, Grønbæk K, Ørntoft T, Waszak SM, Dahan O, Pedersen JS, Lund AH, Pilpel Y (2014) A dual program for translation regulation in cellular proliferation and differentiation. *Cell* 158(6):1281–1292. <https://doi.org/10.1016/j.cell.2014.08.011>

Single-Cell Multiomics: Dissecting Cancer



Janani Sambath, Krishna Patel, Sewanti Limaye and Prashant Kumar

1 Introduction to Tumor Ecosystem and Single-Cell Analysis

Cancer is a heterogeneous and complex disease with its own “tumor ecosystem” where tumor cells interact with neighboring cells which allows them to adapt and evolve continuously [1]. Tumor microenvironment consists of tumor cells surrounded by plethora of other cell types such as fibroblasts, epithelial cells, immune cells, inflammatory cells, blood and vascular networks, as well as extracellular matrix, which broadly defines the tumor microenvironment (TME). Schematic representation of TME is depicted in Fig. 1. In homeostasis, TME acts as a physical barrier against the malignant cells. However, tumor evolution reprograms the adjacent TME to facilitate tumor growth and progression [2]. Cross talk between stromal cells and cancer cells is reported to result in host metabolism hijacking, immune evasion, and eventually metastasis [1]. Active participation of tumor microenvironment in carcinogenesis has laid foundation for new therapeutic strategies by interfering the cross talk between the tumor and its milieu. Here, we have described major component of tumor microenvironment and their potential role in the carcinogenesis.

J. Sambath · K. Patel · P. Kumar (✉)

Institute of Bioinformatics, International Technology Park, Bangalore 560066, India
e-mail: prashant@ibioinformatics.org

K. Patel

Amrita School of Biotechnology, Amrita Vishwa Vidyapeetham, Kollam 690525, India

S. Limaye

Kokilaben Dhirubhai Ambani Hospital and Medical Research Institute, Mumbai 400053, India

P. Kumar

Manipal Academy of Higher Education (MAHE), Manipal, Karnataka 576104, India

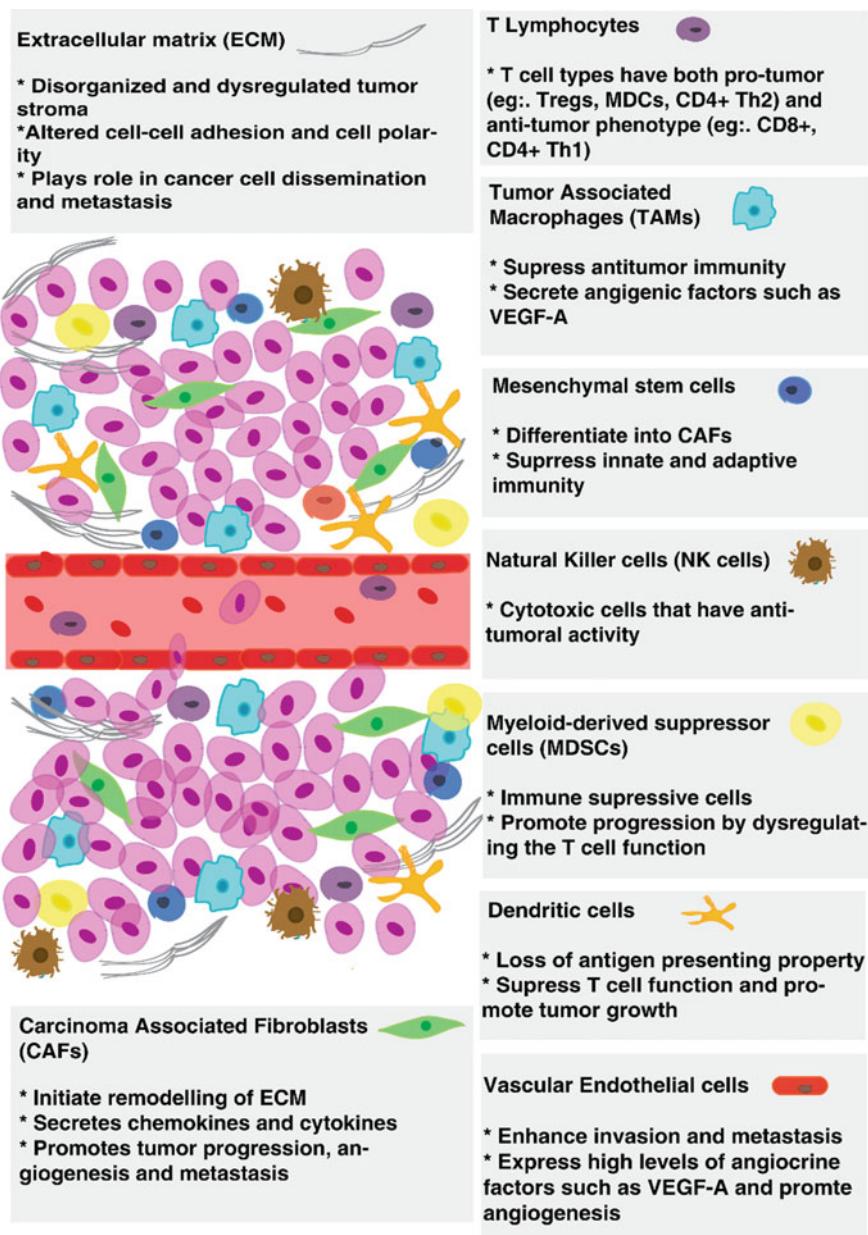


Fig. 1 Schematic representation of tumor microenvironment (TME). TME comprises of stroma, activated fibroblasts, endothelial cells, immune cells, and inflammatory cells such as lymphocytes, MDSCs, dendritic cells, natural killer cells, and TAMs, abundant ECM and vasculature

1.1 Major Components of Tumor Microenvironment and Their Role in Carcinogenesis

1.1.1 Cancer-Associated Fibroblasts

Cancer microenvironment consists of fibroblasts which constitute major subpopulation of stromal components. Tumor cells secrete high level of transforming growth factor β (TGF β), which acts as a chemotactic for fibroblasts and further transdifferentiates them into cancer-associated fibroblasts (CAFs) [3]. The growth factors secreted by tumor cells such as IL-6, IL-1, TGF β , stromal-derived factor 1 (SDF-1 α), and the hypoxic tumor microenvironment recruits mesenchymal stem cells which can differentiate into CAFs, macrophages, or endothelial cells. The transdifferentiation of mesenchymal stem cells contributes to remodeling the shape of tumor microenvironment and promotes tumor growth, angiogenesis, and drug resistance [4, 5]. TGF β has been widely reported to induce the differentiation of mesenchymal stem cells to CAFs in many solid tumors which is a major component in tumor stroma, thereby providing selective advantage to tumor cells [6, 7]. The major hallmarks of cancer comprise of self-sufficiency in growth signals, insensitivity to anti-growth signals, evading apoptosis, limitless replicative potential, sustained angiogenesis, tissue invasion, and metastasis [8, 9]. CAFs enhances tumor progression by remodeling the extracellular matrix (ECM), inducing angiogenesis, restoring inflammatory cells, and directing cell–cell interaction. CAFs are involved directly to influence tumor progression by secreting growth factors and immunosuppressive cytokines. Secretion of SDF1 by CAFs is known to stimulate angiogenesis, and chemokines (CXC motif) ligand 12 (CXCL12) is known to induce epithelial-to-mesenchymal transition in gastric and prostate cancer [10].

1.1.2 Immune and Inflammatory Cells

Immune system plays an important role in recognizing and eradicating pathogenic microbes and other invaders such as tumor cells. In the past decade, studies have reported mutation(s) in the oncogenes such as KRAS and EGFR in lung cancer [11], tumor suppressor genes such as IDH1 in glioma [12], and BRCA1/2 in breast cancer [13] that are widely used as biomarkers for diagnosis, prognosis, and disease management. Activating mutations in tumor suppressor genes and oncogenes alters cytokines, chemokines, and inflammatory mediators in the tumor microenvironment and recruits the inflammatory cells into the milieu [3]. Inflammatory cells induce inflammatory signals and shape the cancer-related inflammatory environment that sequentially boosts the neoplasm progression by circumventing the cancer cells from immune invasion. All immune cells are subjected to polarization, which can exert both antitumor and protumor activity [14]. Among all the immune cells, macrophages comprise majority of population and play crucial role in promoting tumor growth. Macrophages which are polarized into tumor-associated macrophages

(TAMs) elicit pro-tumorigenic activity and enhance tumor invasion and metastasis, angiogenesis, and extracellular matrix remodeling by inhibiting the immune surveillance. Macrophages are recruited into the microenvironment by vascular endothelial growth factor (VEGF), macrophage-colony stimulating factor (M-CSF), and monocyte chemotactic protein 1 (MCP-1) produced by tumor cells [3]. Natural killer cells which have a key feature of recognizing and eradicating malignant cells are limited by the secreted soluble factors such as prostaglandin E2 (PGE2) and TGF β . Myeloid-derived suppressor cells (MDSC) and regulatory T cells (Treg) have an innate function of immune suppression which could be activated by cytokines through signaling pathways [15, 16]. Accumulation of MDSCs is driven by tumor necrosis factor (TNF), and these cells will help the tumor cells escape from the immune surveillance, inhibit adaptive immunity, and promote angiogenesis by secreting VEGF, TGF β , and fibroblast growth factors (FGF) [2, 3].

1.1.3 Blood and Lymphatic Vascular Network

Tumor cells require metabolites and nutrients for their growth which is facilitated by formation of complex vascular network. Tumor blood vessels are formed from pre-existing normal blood vessels, tumor blood vessels, and tumor endothelial cells (TECs). TECs have an altered phenotype from normal endothelial cells and promote angiogenesis by secreting angiogenic factors [17]. TECs show a pattern of chromosomal instability and express distinct expression pattern when compared to normal endothelial cells [18]. The genetic and epigenetic alterations in the TECs aid the process carcinogenesis by interacting with the tumor microenvironment [18]. The vascular network formed around the tumor cells is inefficient and leaky in nature. To overcome the hypoxia condition, the angiogenic switch is activated during the process of tumorigenesis for the formation of new blood vessels [10]. Key blood/lymph-angiogenesis-related factors including VEGFA, VEGFC, and platelet-derived growth factor A (PDGFA) are highly expressed in the tumor cells. The growth factors and their corresponding receptors induce many signaling cascades and contribute to tumor-associated blood/lymph-angiogenesis [19]. Blood and lymphatic networks help tumor cells evade from immune destruction. In particular, MDSC and the immature dendritic cells present in the sentinel lymph node (SLN) will restrict the operation of T cells. SLN will have physical contact with the tumor cells and provide a freeway for tumor cells to migrate into other regions [10].

1.1.4 Extracellular Matrix (ECM)

Extracellular matrix is the non-cellular component of the tumor microenvironment characterized by different physical and biochemical properties. The composition and the function of ECM are unique for each organ. Major components of ECM include collagen, fibronectins, laminin, proteoglycans, and hyaluronans [10, 20]. The ECM in stroma associated with tumor cells is reported to be disorganized and deregulated.

The abnormal ECM found in all stromal cell types secretes numerous growth factors, cytokines, and hormones to escalate tumor invasion and metastasis [21]. As tumor cells proliferate, the surrounding ECM undergoes significant architectural changes in a dynamic interplay between the microenvironment and resident cells. Continuous cross talk between the ECM and the tumor cells promotes tumor progression by disturbing cell-cell adhesion, cell polarity, and inducing growth factor signaling. Collagen and fibronectins provide mechanical strength to the cells preventing the tumor cells from migrating [22]. Cancer cells have adopted strategies to cross these barriers. As the tumor cells proliferate, there will be an increased mechanical stress in the membrane resulting in rupture of the membrane that provokes the tumor cells to escape from the primary site [22]. Understanding the key elements of ECM will guide for new therapeutic interventions for cancer.

1.2 Conventional Sequencing Approaches and Their Limitation to Characterize TME

Advance in genomic techniques has revolutionized healthcare system with the advent of individual's genomic blueprint and emphasized the concept of personalized medicine. Next generation sequencing technologies have catalogued information on alterations in multiple diseases including cancer, inherited disorders, infectious diseases, and so on. Identification of genomic variations in normal and tumor tissues has made notable impact on disease diagnosis, prognosis, and treatment response [23]. However, one of the challenging issues with this approach is the genetic diversity that forms the basis of the intratumor heterogeneity (ITH) [1, 24]. Conventional genomic, transcriptomic, and proteomic studies on bulk tumor have not been able to deduce cellular heterogeneity and complexity. Multiregion whole-exome sequencing associated with bulk sequencing approach has been harnessed to identify ITH of many cancer types; however, it cannot directly dissect the cellular composition of the tumor region [25, 26]. Computational deconvolution method has been employed to analyze gene expression data to infer the cellular composition of tumors with the help of known gene signatures or gene expression profiles for sorted cells [27]. However, this method is limited to known cell types and could not infer cancer and non-malignant cell-type proportions directly from tumor gene expression profiles. Advent of single-cell sequencing technology has a great potential to capture the nature of cancer heterogeneity. The recent advances in single-cell sequencing techniques have indisputably revolutionized the field of cancer research.

1.3 Single-Cell Sequencing Techniques and TME

Single-cell sequencing technique has wide range of options to characterize the tumor heterogeneity. It has given extensive knowledge about heterogenic nature of the cancer, rare subpopulation of cells, clonal evolution, and the development of drug resistance. Single-cell DNA sequencing studies have been applied to explore the clonal and subclonal architecture of the primary tumors and the dynamics of mutation acquisition in breast, bladder, glioblastoma, colon, and hematological malignancies [1]. Single-cell epigenomic technologies have also been utilized to investigate the role of epigenetic alteration in disease progression and metastasis [28].

Bartoschek et al. performed single-cell RNA sequencing (scRNA-seq) on negatively selected mesenchymal cells for CAFs in breast cancer. The study identified three transcriptionally diverse subtypes of CAFs, namely vascular CAFs (vCAF), matrix CAFs (mCAF), and cycling CAFs (cCAF) with unique functional and spatial characteristics [29]. In another study by Li et al., 2 CAF subpopulations (CAF-A and CAF-2) with distinct transcriptome profiles were identified using scRNA-seq in colorectal cancer [30]. The heterogeneity of TAMs in glioma has been studied extensively by Muller et al. using ScRNA-seq. The expression signatures distinguished blood-derived TAMs from brain-resident microglia TAMs and also showed distinct phenotypes. Blood-derived TAMs upregulate immunosuppressive cytokines and correlate with reduced overall survival in lower-grade glioma [31]. ScRNA-seq has been used to profile the expression patterns of the cancer cells to dissect the population of tumor microenvironment and broadly depicted the tumor immune microenvironment in many cancers [32]. These rare subpopulations of cells were identified due to virtue of the in-depth investigation approaches of single-cell sequencing technique.

2 Single-Cell *omics*

Mutational event in tumor suppressor genes and oncogenes is known to aid in the development of cancer. These accumulated mutations over the time confer tumorigenic properties to cells and eventually diverge to form distinct subclones with different mutational landscape. This forms the basis for ITH, which is included as a major hallmark of cancer. Clonal diversity in human tumors plays key role in progression, metastasis, and evolution of resistance to therapy. However, in-depth investigation of ITH was hampered due to technical advancement thus far. Studying clonal diversity in bulk tumor sequencing is difficult because the tumor tissue obtained for sequencing includes mixture of tumor cells resulting in masking of mutations present at low frequency that may have important role in tumor progression. To overcome this challenge, recent studies have investigated tumor profiling at the single-cell level (Table 1). Single-cell sequencing includes single-cell genome, transcriptome, and epigenome and multiomics sequencing led to detailed understanding of tumor

Table 1 List of single-cell sequencing studies

S. No.	Cancer type	Sequencing method	Sample type (cells)	References
1	Pancreatic cancer	scRNA-seq	5403	[133]
2	Breast cancer	scDNA-seq	1293	[75]
3	Triple-negative breast cancer	scDNA-seq, scRNA-seq	DNA: 900 RNA: 6862	[134]
4	Breast cancer	scRNA-seq	45,000	[135]
5	Lung adenocarcinoma	scRNA-seq	52,698	[107]
6	Renal cell carcinoma	scWES-seq	30	[41]
7	Colorectal cancer	scDNA-seq	372	[44]
8	Rectal cancer	scWES-seq	88	[136]
9	Lung adenocarcinoma	scRNA-seq	1800	[137]
10	Chronic myeloid leukemia	scRNA-seq	>2000	[58]
11	Liver cancer	scRNA-seq	5063	[61]
12	Breast cancer	Single nuclei RNA-seq	1800	[129]
13	Bladder cancer	scWES-seq	59	[42]
14	Colorectal cancer	scWES-seq	165	[38]
15	Breast cancer	scRNA-seq	515	[62]
16	Colorectal cancer	scRNA-seq	590	[30]
17	Breast cancer	Targeted	14	[119]
18	Prostate cancer	scRNA-seq	77	[125]
19	Breast cancer	Single nuclei DNA-seq	113	[34]
20	Breast cancer cell line	scDNA-seq and scRNA-seq	21	[138]
21	Metastatic breast cancer	Targeted expression profile	964	[59]
22	Hela cell line	scRNA-seq	40	[139]
23	Pancreatic cancer	scRNA-seq	168	[124]
24	Colon cancer	scDNA-seq	63	[37]
25	Acute lymphoblastic leukemia	Targeted single-cell seq	1479	[40]
26	Glioblastoma	scRNA-seq	430	[56]
27	Prostate cancer	scWGS-seq	8	[140]
28	Prostate cancer	scWES-seq	25	[141]
29	Clear cell renal carcinoma	scWES-seq	25	[36]
30	Muscle-invasive bladder cancer	scWES-seq	66	[39]
31	Breast cancer	scWGS-seq	200	[33]

microenvironment, to delineate ITH and clonal diversity. This method also provides information on rare cell population, which may have an oncogenic effect.

2.1 Single-Cell Genomics

Study by Navin et al. employed scSeq analysis approach to investigate clonal evolution in breast cancer for the first time in 2011. This study has profiled copy number alterations and revealed punctuated model of copy number evolution in which the aneuploid rearrangements occur in early stage and followed by stable clonal expansion in two triple-negative breast cancer patients (TNBC) [33]. Later, Wang et al. applied single-cell whole-exome sequencing (Nuc-Seq) method in two breast cancer patients and showed that the point mutations evolved gradually over time generating extensive clonal diversity and supported the concept of punctuation model in copy number evolution. These studies hint at the potential of using scSeq to delineate clonal evolution and elucidate driver genes mutation with low allele frequency [34]. Single-cell sequence analysis of glioblastoma to explore the clonal diversity reported the convergent evolution of EGFR mutations in different subclones within a tumor [35]. In case of renal carcinoma, single-cell genomic analysis revealed that the cells share common truncal mutations indicating their common origin and presence of fewer rare mutations [36]. Study by Yu et al. used single-cell exome sequencing approach and depicted biclonal origin model with the verification of rare mutation event of *SLC12A5* and its potential oncogenic effect in colon cancer [37]. In contrast, Wu et al. using combined bulk whole-exome sequencing and single-cell whole-exome sequencing demonstrated that the colorectal cancer has the monoclonal origin; however, several subclones were found based on the accumulation of novel driver mutations [38]. Genetic diversity of bladder cancer at single-cell level has been investigated by Li et al. They employed single-cell whole-exome sequencing on muscle-invasive bladder transitional cell carcinoma and identified the existence of tumor subpopulation within a tumor. Authors observed set of 22 common mutant genes including some driver genes in all three identified subclones indicating the origin from common ancestral clone [39]. DNA single-cell sequencing (DNA SCS) helped to delineate the clonal evolution in hematopoietic malignancies. Gawad et al. performed targeted sequencing in 1479 cells from six childhood acute lymphoblastic leukemia (ALL) patients to measure the clonal structures [40]. Cancer stem cells play a crucial role in the initiation of tumorigenesis, progression, metastasis, and drug resistance. Single-cell sequencing can be used to characterize the cancer stem cells and to identify the driver genes. Li et al. discovered that novel mutations in KCP, LOC440040, and LOC440563 act as a driver in renal cancer stem cells. The authors concluded that the above three novel mutations can promote the reprogramming of renal cancer cells into cancer stem-like cells (CSCs) [41]. Single-cell sequencing on bladder cancer stem cells (BCSC) deciphered the genetic basis and stemness-related gene mutations in BCSC [42].

Emergence of metastasis remains to be a key characteristic of cancer progression. Studies have demonstrated that metastasis in cancer patients is carried out by specific subclones with unique properties [43]. Single-cell sequencing enabled the researchers to investigate and characterize these rare cell populations. For example, single-cell DNA sequencing approach has been used to trace the metastatic lineages in two colorectal cancer patients with matched liver metastases. The study reported monoclonal metastatic seeding in one patient and polyclonal in another. The authors inferred late-dissemination model of metastasis, in which the primary tumor cells acquire many mutations and CNAs over the time and then disseminate to distant organ sites [44]. Clonal diversity has been studied using animal models like xenografts and genetically engineered mice (GEM). Recently, single-cell sequencing study on mouse xenografts from TNBC patients showed extensive selection of tumor clones during the first few passages of the tumors to other recipients in response to new stromal environment [45].

Aforementioned studies employed single-cell cancer sequencing to decipher the clonal diversity and intratumor heterogeneity. Understanding the tumor at single-cell level will unravel the complex biological process including progression, metastasis, and mechanism of therapy resistance.

2.2 *Single-Cell Epigenomics*

Epigenetics is defined as the heritable change in gene expression without altering the genetic code of DNA. The epigenetic alterations broadly include changes in DNA methylation, and histone modification [28]. These epigenetic events are dynamic and reversible. Along with genetic alterations, epigenetic aberrations also play a crucial role in carcinogenesis. In cancer cells, the pattern of epigenetic changes differs from cell to cell leading to heterogeneity within tumor [28]. Epigenetic alterations in cancer generally lead to silencing of tumor suppressor genes, activation of oncogenes, and dysregulation in gene expression [46]. DNA methylation in the CpG island of gene promoter region results in silencing of gene expression by preventing the binding of transcription machinery proteins. Several key genes were observed to be disrupted in various cancer types either by hypermethylation or by hypomethylation [47]. Chromatic structural changes influenced by histone modifications are another important regulatory mechanism for gene regulation and carcinogenesis. Mutation in the chromatin remodeling factors and histone-modifying enzymes leads to change in the transcriptional state of the gene, thus altering the behavior of the cell [48]. Recent studies have shown that the dysregulation in the epigenetic machinery of a cell can drive the oncogenic transformation, followed by progression and treatment response [48–50].

Advances in the next generation sequencing technologies such as bisulfite sequencing for analyzing DNA methylation, DNase-seq, and MNase-seq for DNA accessibility and chromatin conformation, and chromatin immunoprecipitation followed by sequencing (ChIP-seq) for binding sites of individual factors or modified

nucleosomes [51] presented new opportunities to study the epigenetic regulation at genome-wide scale and also provided the efficient way for integrated analysis of genetic alterations and epigenetic aberrations that contribute to the neoplastic development. However, all these studies involved the average measures of bulk cellular populations revealing the generalized epigenetic features and failed to detect single-cell epigenomic cell-to-cell variability. This limitation in the bulk epigenetic profile can be subdued by the development of single-cell methods to measure the epigenetic variation in the individual cell level.

Epigenetic regulation studied at single-cell level will unmask the epigenome ITH in cancer cells. DNA methylation at cytosine residues (5mC) which is a major type of DNA modification is profiled using bisulfite sequencing (BSseq) method. The first single-cell-based method developed to measure the 5mC within an individual cell was single-cell methylome analysis technique based on reduced representation bisulfite sequencing (scRRBS) [52]. Recently, scRRBS method has been applied to study the accumulation of epimutation in B cells from healthy and chronic lymphocytic leukemia (CLL) cohorts. Even though the epimutation rate is higher in CLL population than normal, authors observed low variability in cell-to-cell epimutation rate in CLL patients. In normal B cells, the variability of epimutation rate diversified over the time of B cell differentiation, serving as an evolutionary molecular clock. Integration of epigenetic alterations with transcriptional profile evidenced the negative correlation between promoter DNA methylation and gene expression. Tree topology showing distinct subclones has been validated by integrating genetic, epigenetic, and transcriptional information [53]. Farlik et al. developed whole-genome bisulfite sequencing (WGBS) assay to profile DNA methylation in small-cell population (μ WGBS) and in single cells (scWGBS) and applied this method to study epigenetic cell-state dynamics in mouse and human cells [54]. Histone modification mapping in single cells is achieved by developing scChiP-seq method; however, the coverage is very low and thus needs to assess large number of cells [28]. A study by Buenrostro et al. employed single-cell assay for transposase-accessible chromatin (scATAC-seq) method to assess the chromatin accessibility changes in K562 leukemic cells. K562 leukemic cells showed cell-to-cell variation in chromatin accessibility by the combined activity of trans-factors in GATA motif. The heterogeneous expression of GATA1 and GATA2 transcriptions factors induced this variation in the cells. Study of variation in chromatin accessibility will help to understand the regulatory mechanisms at single-cell level [55].

Development of single-cell epigenetic techniques will help to seek the hidden epigenetic alterations that are involved in tumor progression. Multiomics approach of integrating epigenome with genomic variation and dysregulated transcript expression will give multilayer information and molecular connections in disease.

2.3 Single-Cell Transcriptomics

Single-cell genomics has led to the identification of detailed mutation portrait in different subclones of the tumor. Expression profile of these cells with distinct genetic makeup can be further investigated using single-cell transcriptomics approach. Single-cell RNA sequencing has provided unprecedented insights into tumor heterogeneity and unraveled the mysteries about cell-to-cell variation, new cell types in the tumor milieu, plasticity, and cancer stem cells. The pioneer study on single-cell RNA sequencing was done in 430 single cells isolated from five primary glioblastomas to uncover the genetic heterogeneity. Authors examined the subtype signatures established by the cancer genome atlas (TCGA) in individual cells. Different glioblastoma subtypes were observed within individual cells demonstrating ITH. The analysis revealed that tumors contain hybrid cell states including progenitor states and differentiated states with distinct transcriptional programs and provide inferential evidence for dynamic transitions [56]. Another landmark study analyzed 4645 single cells from 19 melanoma patients for profiling malignant and non-malignant cells (immune, stromal, and endothelial cells). The study identified the presence of rare subpopulation with drug-resistant property. Single-cell gene expression analysis of infiltrating T cells revealed potential biomarkers for distinguishing exhausted and cytotoxic T cells that may aid in selecting patients for immune checkpoint blockade. This study has unraveled the tumor microenvironment in melanoma that provided insights into targeted and immuno therapies [57]. Giustacchini et al. analyzed about 2000 single cells from chronic myeloid leukemia (CML) patients to characterize distinct molecular signatures of single-cell subpopulations in human CML samples from diagnosis through remission and disease progression. The study revealed heterogeneity of CML single cells and identified subgroup which was persistently found even after prolonged treatment [58].

Tumor microenvironment plays a major role in shaping the tumor at both primary and metastatic sites. It provides an advantageous environment for the cancer cells to evolve and promote its growth. Single-cell RNA sequencing can shed light on tumor microenvironment by dissecting the malignant cells, stromal cells, and immune cell within the tumor. A study by Li et al. demonstrates a single-cell profiling of the cells isolated from primary colorectal cancer tumor and matched normal samples. The study has identified seven cell types, namely epithelial cells, fibroblasts, endothelial cells, B cells, T cells, mast cells, and myeloid cells within tumor. The specific gene expression signature of these cell types has shown the increased level of EMT-related genes in CAFs that could be the cause for transition to activated fibroblast state. Authors proposed that single-cell transcriptomics followed by cell-type clustering provides a more unbiased approach for identifying dysregulated genes within tumor and stromal cells [30]. A study on metastatic breast cancer cells by Lawson et al. reported that the metastatic cells from the low burden tissue possess distinct gene expression signatures for functions like stem cells, epithelial-to-mesenchymal transition, pro-survival, and dormancy when compared with metastatic high burden tissue which possessed properties similar to primary tumors. This study supported

hierarchical model for metastasis, in which metastases are initiated by stem-like cells that proliferate and differentiate to produce advanced metastatic disease [59].

To escape from immune surveillance, tumor cells recruit immunosuppressive cells into the microenvironment and alter the phenotype and function of normal immune cells [60]. Transcriptome sequencing of bulk tissue has classified tumor subtypes based on the expression signature of bulk tumor population. However, it could not classify the cellular diversity. Studies focusing on immune landscape of the tumors have been flourished using scRNA-Seq. Characterization of immune cells found in stromal population will lead to the identification of new immunotherapeutic strategies. Zheng et al. have catalogued immune microenvironment of lung adenocarcinoma and provided the immune atlas of lung adenocarcinoma. In this study, Tregs and exhausted CD8+T cells are found to be more abundant in tumor cell than in the normal peripheral blood, and it can result in immune suppression in the TME [61]. Transcriptome analysis of 515 single cells from different subtypes of breast cancer patients revealed carcinoma and non-carcinoma microenvironment cells that included macrophages, T lymphocytes, and B lymphocytes. Profiling of infiltrating T cells revealed that the infiltrating immune cells were high in TNBC subtype than the other subtype. It showed the signature of T cells with a regulatory or exhausted phenotype and macrophages with an M2 phenotype that can promote evasion of cancer cells from immune surveillance [62].

The studies mentioned here have demonstrated the potential of scRNA-seq in dissecting tumor heterogeneity. scRNA-seq study will facilitate better understanding of individual cell types within tumor and their role in carcinogenesis. This information can have enormous impact on clinical research and on the precision medicine. Similarly, multiple single-cell omics analysis studies have been reported in various cancers to delineate ITH.

3 Cancer: Dissecting Tumor Heterogeneity

Tumor heterogeneity is characterized by the presence of multiple subclones that evolve during the process of carcinogenesis. During this evolutionary process, subclones evolve by acquiring various changes in genomic, transcriptomic, and epigenome level among others and continuously undergo selection pressure similar to Darwinian natural selection. The presence of these subclones is known to reduce the efficacy of treatment by acquiring resistance and potential promotion of metastasis. To delineate ITH, deep sequencing, multiregional sequencing, and single-cell sequencing are predominantly employed. Using computational methods such as PyClone [63], Sci-Clone [64], and GenoClone [65], subclonal architecture can be inferred based on mutational allele frequency (MAF) from whole-exome sequencing of the bulk tumor. Multiregion sequencing has unveiled the presence of spatial intratumor heterogeneity in many cancer types.

3.1 Multiregion Sequencing and Tumor Heterogeneity

Multiregion sequencing in localized lung adenocarcinoma is reported by Zhang et al. to characterize ITH. Phylogenetic tree representing evolution trajectory of genomic alterations depicts that 76% of mutations and 22 known cancer genes mapped to trunks. Hence, it is hypothesized that these mutations were acquired at early stage of carcinogenesis. After treatment, three patients have relapsed and these individuals had higher frequency of subclonal non-trunk mutations in the primary tumors compared to patients without relapse, suggesting that the presence of subclonal mutations in higher frequency likely increases the risk of relapse [66]. Previous study was employed using all somatic mutations into consideration; however, some studies have focused only on somatic mutations in driver genes. Zhang et al. performed multi-region whole-exome sequencing in 12 *TP53*-driven non-small-cell lung adenocarcinoma patients. In concordance with other studies, phylogenetic tree demonstrated the model of branched evolution during tumor development. In classical tumor biopsy although tumors are classified as *TP53* mutated, different region of samples consisted of both *TP53* mutated and wild-type tumor cells. Similar pattern has been observed in other known cancer genes such as *CDKN2A* and *TTN* [67].

Recent intratumor mutational heterogeneity study on head and neck squamous cell carcinoma (HNSCC) of different primary sub-sites such as larynx, floor of the mouth (FOM), and oral tongue has reported varied degree of ITH. Laryngeal and FOM tumors have high degree of ITH compared to oral tongue tumor, suggesting that the degree of ITH varies with different primary tumor sub-site in HNSCC [68]. Studies using multiregion whole-exome sequencing (M-WES) in esophageal squamous cell carcinoma (ESCC) reported that driver mutations in oncogenes and tumor suppressor genes were mapped as both truncal and branched in the phylogenetic tree indicating the early and late events in the tumor evolutionary process. Copy number alterations in the genes such as *TP53*, *CDKN2A*, *CCND1*, and *EGFR* showed significant spatial ITH in ESCC [69, 70].

M-Seq of primary and liver metastatic regions from five colorectal patients profiled somatic mutations and copy number alterations and classified them under five spatial categories (universal, metastatic-clonal, primary-private, metastatic-private, and unclassified). The study reported 19.8–53.9% of mutations as universal, 46.1–80.2% of mutations as subclonal, and 1.4–37.2% as unclassified. Mutational allele frequency observed to be higher in metastatic tumor than primary tumor evidencing the event of evolutionary bottleneck. Phylogenetic trees inferred branched evolution pattern in which clones diverge from a common ancestor and evolve in parallel in the tumor mass resulting in multiple clonal lineages [71]. Wei et al. characterized intratumor heterogeneity in colorectal cancer using multiregion WES approach of matched primary and metastasis tumors and reported polyclonal seeding mechanism for metastasis where multiple subclones are hypothesized to have disseminated from primary tumors [25].

3.2 Clonal Expansion Models

Rapid developments in the next generation sequencing techniques made possible to uncover alterations in genome of an individual sample, and the development of bioinformatics algorithm helped to delineate the clonal lineages and hypothesize the clonal evolution model in various cancer types. Single-cell sequencing has improved our understanding of tumor evolution. Various clonal expansion models have been proposed in early stage as well as advance stage cancers. Detailed understanding of such evolution model could provide us insight into step-by-step progression of cancer through various stages.

Various expansion models hypothesized based on genomic alteration are reported in cancers such as prostate, colorectal, breast, lung, and bladder. In case of castration-resistance prostate cancer (CRPC), two models have been proposed to understand the clonal evolution, i.e., adaption model and clonal selection model. Adaption model proposes that the early stage of tumor is in native state comprising of androgen-dependent cells. In the androgen-deprived environment, these cells undergo genetic/epigenetic alteration to adapt the environment resulting in the emergence of androgen-independent cells also referred as castration-resistance cells. Clonal selection model suggests that the tumor is composed of heterogeneous cells including androgen-dependent cells and castration-resistance cells. In an androgen-deprived environment, these resistance cells are selected for their survival and progression, whereas the androgen-dependent cells are swiped off [72]. Recent scRNA-seq study evidenced the presence of minor subpopulation of androgen-independent cells which are positively selected after the androgen-deprivation therapy proposing clonal selection model (Fig. 2a) in prostate cancer [73].

Ductal carcinoma in situ (DCIS) which is a most prevalent early form of breast cancer rarely progresses into invasive ductal carcinoma (IDC). The atypical progression from DCIS to IDC raises clinical challenges in diagnosis and treatment of the patients. Next generation sequencing studies have reported intratumor heterogeneity with clonal architecture and evolution in invasive breast cancers. Three invasion models, i.e, independent lineage model, evolutionary bottleneck model, and multiclonal invasion model, have been proposed to understand the progression from DCIS to IDC based on genomic alteration pattern. Independent lineage model proposes that the two independent initiating cells from normal breast tissue give rise to separate DCIS and IDC subpopulation, whereas evolutionary bottleneck model hypothesizes that single normal cell in breast tissue give rise to both DCIS and IDC subpopulations. Evolutionary bottleneck posits that several clones evolve in the ducts and one selective clone migrates and forms invasive cancer. In contrast, multiclonal evolution hypothesized that multiple clones evolved in the duct escape the basement membrane and establish invasive carcinoma. Genomic studies on bulk tumor could not distinguish precise clonal evolution. Some of these studies supported evolutionary bottleneck model, indicating that the single normal cell gives rise to DCIS and IDC subpopulation [74]. Recently, topographic single-cell sequencing study of 1293

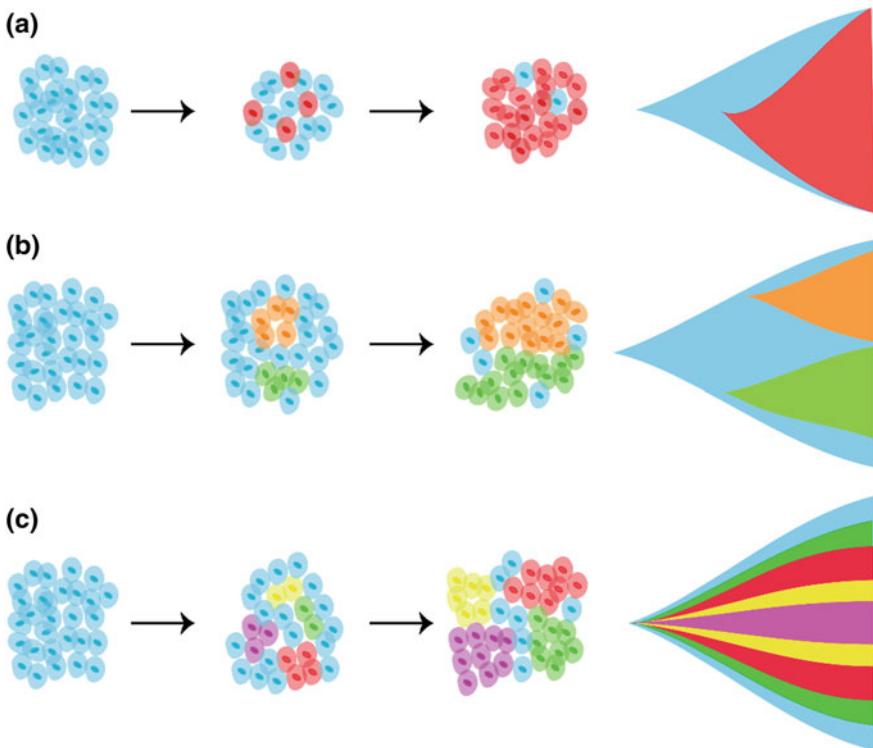


Fig. 2 Schematic representation of clonal evolution models exhibited by various cancers. **a** Clonal selection model. Subclones that have acquired additional mutations are positively selected and provide favorable TME. **b** Multiclonal invasion model. Multiple clones evolve simultaneously and contribute equally to the growth of tumor. **c** Big Bang model. Tumor growth will occur in the absence of stringent selection, consistent with neutral evolution

cells from 10 patients with both DCIS and IDC regions revealed that genomic alterations in the subclones are present before it leaves the duct and hypothesized that all the subclones in the duct arise from the single initiating cell. The evident shift in subclonal population frequency between DCIS and IDC supports multiclonal invasion model (Fig. 2b), in which multiple subclones escape from the duct and evoke the invasive tumor [75].

In case of colorectal cancer, one clonal expansion model, i.e., Big Bang model (Fig. 2c) is proposed which states that single tumor initiation cells lead to the propagation of intermixed cells with various subclones without stringent selection, consistent with effective neutral evolution. Analyzing the genomic profiles of multiregion, single gland and single cells from 15 colorectal tumors exhibited uniform high intra-tumor heterogeneity and also evidenced the implications of Big Bang model. In this model, public driver alterations and private alterations that are pervasive in the final neoplasm might have occurred early during tumor growth. Big Bang model also insists that the timing of mutation determines its prevalence in the advanced

tumor. Hence, the private alterations which occur early will be found in subclonal populations, whereas late mutations will become undetectable [76].

Study of clonal evolution which drives the tumor progression will provide detailed information about the initial stage of cancer development and help us to treat cancer efficiently. During cancer progression, each tumor cell is reported to undergo various selective pressures that form the basis of intratumor heterogeneity. Each tumor will exhibit different patterns of evolution, and delineating the model of evolution will give novel insights into the progression and identification of subpopulations that drive metastasis or resistance to anti-cancer therapy will provide new therapeutic targets and improve treatment efficiency.

4 Single-Cell Analysis: Drug Resistance Mechanisms

Cancer drug resistance mechanisms are complex and multidimensional. Cancer therapies are known to become resistant over a period leading to recurrence and disease progression. As the realization of rising rates of cancer therapy resistance grew, there was a pressing need to understand the drug resistance mechanisms better. Several different mechanisms of cancer drug resistance have been discovered and described (Fig. 3) [77–80]. Drug efflux, drug inactivation, drug target alteration, tumor cell death inhibition, tumor heterogeneity, DNA damage repair, epithelial-to-mesenchymal transition, epigenetic modifications, and multidrug resistance were widely studied mechanisms associated with resistance. These mechanisms could be present in isolation or in combination leading to the modality of combination therapy for most cancer diagnosis [77–82]. The understanding of these mechanisms for the mainstay of development of strategies to design personalized therapeutic approaches for these patients would be utmost important.

Nearly two-thirds of non-small-cell lung cancer (NSCLC) patients are diagnosed at an advanced stage where surgical resection may not be possible. These patients need systemic therapy and in some cases radiation-based therapy. Disease relapse and progression is a common phenomenon in NSCLC and warrants a deeper understanding of the pathways of resistance for better implementation of care, decreased morbidity and mortality in this deadly disease [81]. Head and neck squamous cell cancers (HNSCC), especially the non-HPV type are the other type of cancers that are known to have high risk of recurrence and disease progression [82]. Oral squamous cell carcinoma (OSCC) is one of the commonest forms of head and neck cancer and is known to be difficult to treat with high relapse rates. Nearly sixty percent of HNCSCC are locally advanced at diagnosis and are not resectable. These patients as well as resected high-risk cases are treated with chemotherapy and radiation in the definitive or adjuvant setting. Resistance to chemotherapy and radiation is one of the prominent reasons for relapse and progression in these cases [83].



Fig. 3 Drug resistance mechanisms in cancer. Various mechanisms promote drug resistance either directly or indirectly, independently or in combination

4.1 Mechanisms of Drug Resistance

4.1.1 Drug Efflux, Drug Inactivation, and Drug Target Alterations

Several mechanisms of drug resistance have been studied and described [77–81]. Drug efflux is one of the most prominent mechanisms of resistance leading to decreased efficacy, high relapse, and decreased survival. ATP-binding cassette (ABC) transporter proteins are regulatory proteins on the plasma membranes of cells that are understood to enable this efflux [84]. The three transporters that are thought to be notorious for resistance in cancer are multidrug resistance protein 1 (MDR1), multidrug resistance-associated protein 1 (MRP1), and breast cancer resistance protein (BCRP) [85, 86] and are known to be associated with poor clinical outcomes [86]. The ABC genes encoding MDR1, MRP1, MRP2, and BCRP are involved in many solid tumor chemotherapy resistances including OSCC and NSCLC [87].

Decreased drug activation and drug inactivation are other prominent mechanisms of cancer drug resistance. The mechanism usually involves complex processes in which drugs interact with different proteins leading to their modification, partial or full degradation, or inactivation. Matrix metalloproteinases (MMPs) overexpression has been also associated with drug resistance in HNSCC [88]. Endothelial growth

factor receptor (EGFR) overexpression seen in HNSCC is linked with response from EGFR-directed therapy in this disease. Competitive inhibition of EGFR blocking antibodies has been shown to lead to inactivation of EGFR inhibitors and resistance to therapy. Several other prominent protein superfamilies are involved in altered drug activation and inactivation like uridine diphospho-glucuronosyltransferase (UGT) which is involved in irinotecan inactivation modulated by epigenetic silencing of UGT1A1 in colon cancer, glutathione-S-transferase (GST), and cytochrome P450 (CYP) superfamily [89].

Drug target alterations have major implications in the development of resistance. Alterations in target receptors and expression levels could be one of the other mechanisms for cancer drug resistance. A classic example of target receptor alteration would be the development of T790M mutation in epidermal growth factor receptor (EGFR)-positive non-small-cell lung cancers seen in more than 50% of patients over a period of time leading to drug resistance to first- and second-generation tyrosine kinase inhibitors [90]. Another common example would be the development of resistance to HER2 inhibitors in HER2-positive breast cancer due to the occurrence of PI3KCA mutation or PTEN loss or both in these patients [91].

4.1.2 Inhibition of Cell Death and DNA Damage Repair

Cell deaths by apoptosis or autophagy are common mechanisms of drug-induced cancer kill. However, resistance to apoptosis or autophagy could lead to drug resistance. One of the areas of drug development is trying to address prevention of cell death inhibition by inhibiting autophagy-dependent resistance. For example, hydroxychloroquine is being used to inhibit autophagy and restore estrogen receptor sensitivity for ER pathway inhibitors like tamoxifen in hormone receptor-positive metastatic breast cancer [92]. Alternatively, cellular mechanisms like DNA damage repair (DDR) could reverse drug-induced DNA damage leading to cancer cell repair and resistance to therapy. As an example, one of the most prominent paths of resistance to cisplatin is due to DNA damage repair leading to increase rates of relapse and disease progression [93]. DNA damage repair mechanisms are one of the prominent reasons for resistance to cisplatin-based therapy in HNSCC and may be caused by nucleotide excision repair (NER and BER) gene complexes [94].

4.1.3 Epithelial-to-Mesenchymal Transition and Epigenetic Alterations

Researchers are also studying the role of epithelial-to-mesenchymal transition (EMT) in cancer drug resistance. EMT is one of the complex mechanisms by which tumor cells decrease the expression of cell adhesion receptors responsible for cell–cell attachment while increasing the expression of cell motility adhesion receptors. The degree of EMT and the level of differentiation are thought to correlate with the degree of drug resistance. Studies have reported that resistance to immunotherapy

has been associated with immune exclusion and EMT in melanoma, lung adenocarcinoma, and squamous cell carcinoma [95]. Other mechanisms that included DNA methylation and histone acetylation are the epigenetic modifications that contribute to drug resistance. DNA demethylating agents and histone deacetylators have been investigated in conjunction with chemotherapy and other targeted therapy drugs in an effort to reverse resistant pathway development. Early findings are promising although additional work is needed to bring the strategies for the clinic utility [96].

4.1.4 Heterogeneity of Cancer Cells

Cancer cell heterogeneity has been studied widely and remains one of the biggest challenges in understanding disease biology and formulating therapeutic strategies. Proliferation and growth of drug-resistant clones alongside response to therapy in drug-sensitive clone of cancer cells are well-understood phenomena [97]. Weak selection pressure or state of neutral evolution is thought to be largely responsible for intratumoral heterogeneity. Under weak selection pressure or neutral evolution, multiple different cancer cell clones survive that leads to drug resistance and increased metastatic potential. This condition is specifically described in certain solid tumors [98].

4.1.5 Resistance to Immunotherapy

Immunotherapy with immune checkpoint inhibitors has led to enhanced survival in multiple solid tumors including melanoma, HNSCC, and lung cancer. Resistance to immune checkpoint inhibitors is a complex process and an interplay between the tumor cells, immune cells, and the tumor microenvironment (24). Primary resistance to therapy or secondary resistance with treatment failures may happen during the course of therapy. High PD-L1 expression and high tumor mutation burden have correlated with higher neoantigen load and higher response to therapy with CTLA-4 inhibitors and PD-1, PD-L1 inhibitors in melanoma and lung cancers [99, 100]. Low tumor immunogenicity is thought to be responsible for resistance to therapy with checkpoint blockade and has been demonstrated in pancreatic and prostate cancers. Neoantigen loss has been shown to be responsible for immunoediting and secondary resistance to immunotherapy. Alterations in tumor microenvironment and cell signaling pathways lead to differential immune response and contribute to treatment resistance [101].

4.2 Single-Cell Genomics and Transcriptomics to the Rescue of Drug Resistance

Single-cell multiomics with the help of next generation sequencing and modern computational bioinformatics analyses has provided for better understanding of cancer biology and pathology. Such studies would be critical in cancer-related signaling recognition and the development of novel therapeutics. Initially published by Tang et al. [102], the techniques have improved over time with significant advancement in cell capture methods and library preparation techniques [103, 104]. Genetic heterogeneity has been studied widely with the help of sequenced datasets with single-cell whole-genome and whole-exome sequencing [105]. With improved techniques of single-cell sequencing, it has become easier to study neoplastic clonal populations with better understanding [36]. Single-cell multiomics has enabled to study the transcriptome, genome and methylome of a single cell [106]. Multiomics study by Hou et al. utilized the technique to identify two different clones of hepatocellular carcinoma cells with distinct population and metastatic potentials [106]. These techniques have been also utilized to understand TME and pathways of immunotherapy resistance in lung cancer [107]. Inter-tumor and intratumor heterogeneities have been thought to be one of the causes for partial response to immunotherapy. In a recent study by Ma et al, single-cell RNA sequencing was utilized to confirm heterogeneity of immune response-related genes in lung adenocarcinoma, demonstrating that how modern technology could confer associated drug resistance mechanisms [108].

5 Circulating Tumor Cells: Significance of Single-Cell Analysis

Circulating tumor cells (CTCs) are cancer cells that shed from primary tumor and disseminate into bloodstream. CTCs are the major prerequisite in the metastasis cascade. During cancer progression, tumor cells intravasate into blood vessel by undergoing epithelial-to-mesenchymal transition (EMT). These circulating cells leave the circulation and seed in secondary site via mesenchymal to epithelial transition (MET) and initiate metastasis [109–111]. Studies have mainly focused on CTC isolation and enrichment methods and observed the correlation between CTC number and clinical prognosis [110, 112–114]. Role of CTCs as biomarkers for diagnosis, prognosis, and monitoring treatment response has been reported in several solid tumors [115–117]. Mutation and expression studies of primary and metastatic tumor fostered the advent of personalized medicine. However, the sequencing efforts made are predominantly derived from the single-site biopsy samples that give genomic snapshot of tumor at one time point. The continuous evolution of tumor cells reflects inter- and intratumor heterogeneity in the individual patient. Single-site biopsy samples could not uncover the process of tumor evolution, and subjecting the patient to multiple site biopsies is infeasible. CTCs being a representative of primary and metastatic

tumor cell populations could be used to study tumor heterogeneity and therapeutic resistance mechanism in pan-cancer. The development of single-cell sequencing techniques has facilitated characterization of these rare cancer cell populations.

Recent studies on single CTCs have given insights into the tumor heterogeneity in various cancer types. Studies on single CTCs by Shaw et al. and Luca et al. highlighted the existence of genetic heterogeneity within and between breast cancer patients by observing somatic mutational status of cancer-related genes (*TP53*, *ERBB2*, *EGFR*, *PIK3CA*, *PTEN*, *ESR1*, *KRAS*) [118, 119]. Gao et al. analyzed copy number alterations (CNA) in CTCs isolated from four different types of cancer and proposed convergent evolutionary model for the evolution of tumor CNAs. The proposed CNV evolution model is in contrast to the reported model such as gradual evolution and punctuated model [120]. Ni et al. employed genomic analysis of single CTCs in lung cancer patients and identified that copy number variations in the cancer-related gene loci are the key events that alter the gene expression and give selective advantage for metastasis [121]. Small-cell lung cancer (SCLC) is an aggressive disease with early metastatic dissemination. Major challenge in SCLC is the early development of resistance to chemotherapy. Since SCLC is characterized by early vascular dissemination liquid biopsy sample CTC can be used for diagnosis and predicting clinical outcomes. Carter et al. generated whole-genome sequencing data for single CTCs, pooled CTCs, and matched WBC isolated from 13 SCLC patients and carried out copy number variation analysis. Genes that are altered in CTCs are consistent with tumor cells and are not found in germline control. This result suggested that the CTC-based genomic profile could be an alternative for tumor profiles. Authors used CTC-based CNA data to distinguish patients as chemosensitive and chemorefractory. Using bioinformatics analysis, 16 CNA profiles were generated and developed predictive model that stratifies the samples as chemosensitive and chemorefractory. The study suggested the use of non-invasive CTC-based profiling to anticipate clinical outcomes in SCLC and showed the potential of CTCs to understand the biology of cancer [122].

To understand the mechanism of EMT in cancer metastasis, Yu et al. characterized breast cancer CTCs and suggested that mesenchymal CTCs are highly associated with disease progression [123]. Ting et al. performed single-cell RNA-seq study on CTCs and matched primary tumors from mouse model of pancreatic cancer. Clustering based on transcriptomic profiles of CTCs, primary tumors, and tumor-derived cell lines showed CTCs as separate cluster featured with low proliferative signature, expression of both epithelial and mesenchymal markers, and enriched stem cell-associated genes. Authors observed aberrant expression of extracellular matrix genes, underlying the role of stromal microenvironment in spreading of cancer to distant sites [124].

Gene expression study on individual CTCs of castration-resistant prostate cancer patients showed high intercellular transcriptional heterogeneity compared to expression profile of single cells of prostate cancer cell lines and primary tumor. The authors also observed different androgen receptor splice variants and androgen receptor mutations in individual patients expressing remarkable heterogeneity. The differences in the transcriptomic profile of CTCs from treatment-resistant and treatment-naïve

patients resulted in the identification of several signaling pathways which may confer resistance to anti-androgen therapy [125]. Grillet et al. established CTC cell lines from metastatic colorectal cancer patients characterized by cancer stem cell (CSC) phenotype and genetic and phenotypic heterogeneity. RNA-seq study of these cell lines showed high-level expression of CSC markers and demonstrated the ability of CTCs to develop tumor in distant organs. Expression analysis also showed enrichment in drug-metabolizing pathways which may confer resistance to conventional cytotoxic compounds. This study demonstrated the potential use of CTCs to predict drug responses in colorectal cancer patients [126].

Technological advancements in CTC isolation and sequencing have opened a new dimension of research to use CTCs as biomarker for diagnosis, to predict treatment response, and to study tumor heterogeneity. The pioneer studies on single-cell sequencing of CTCs have successfully demonstrated the use of CTCs for clinical purposes. Further development in the field of CTCs may help to understand tumor heterogeneity and treatment resistance more deeply and facilitate precision medicine with the identification of new therapeutic targets and therapeutic strategies.

6 Limitations and Future Directions

Single-cell sequencing is a giant leap in the field of cancer research. Study at single-cell level has given deeper insights into the biology of cancer, intratumor heterogeneity, clonal diversity, evolution of therapy resistance, metastatic dissemination, and characterized rare cells in the tumor microenvironment. Thus far, huge collaborative efforts such as The Cancer Genome Atlas (TCGA) and International Cancer Genome Consortium (ICGC) have made to collate genomic, epigenomic, transcriptomics and proteome alteration profile in various cancers. Profiling in these cases was predominantly employed using bulk tumor sequencing; hence, these studies could not provide resolution and deeper understanding of ITH. The development of the sophisticated algorithms such as PyClone [63], Sci-Clone [64], and Geno-Clone [65] which exploit variant allele frequency and genotype to delineate ITH is now extensively used to analyze existing sequencing datasets. Recent pan-cancer ITH study using 3383 whole-exome datasets belonging to nine different cancers from TCGA revealed that breast, urothelial, head and neck, and renal carcinoma had relatively lower ITH. These were dominated by single clonal expansion. However, lung squamous cell carcinoma, lung adenocarcinoma, glioma, prostate cancer, and melanoma were predicted to be polyclonal and have higher ITH. This study has efficiently demonstrated the use of open-source tools to investigate ITH in conventionally sequenced bulk tumors. Multiregion sequencing of a tumor biopsy is also a promising approach to study ITH. This technique is predominantly employed in clinical trials such as “Tracking the Evolution of Non-Small-Cell Lung Cancer” also referred to as TRACERx. This study has utilized multiregion sequencing along with orthogonal bulk tumor sequencing to investigate effects of ITH on tumor progression and overall survival outcome based on mutation evolution trajectories.

It is evidenced that the cancer is spatially and temporally heterogeneous in nature. One of the major drawbacks of single-cell sequencing is the loss of spatial information during isolation process. During the process of single-cell sequencing, cells are segregated using methods such as FACS [127, 128], nanowells [129], microdroplets [130], or micromanipulation [131]. Hence, crucial information about spatial distribution of subclones in tumor samples is lost in the process of cell preparation leading to loss of information about their organization and migration pattern. Recently, topographic single-cell sequencing method has been developed to restore the spatial information of cells during isolation. Single-cell sequencing has added an additional dimension to existing knowledge of genomics, epigenomics, and transcriptomic alterations in tumors. However, this technique should be utilized with caution as reports suggest problems such as high error rates, low throughput, and low-sequencing efficiency [132].

Due to technological and economic limitations, hundreds to thousands of cells only sequenced from tumor biopsy. Hence, there are fair chances to miss out the rare cell subpopulation that may have the potential to drive tumor progression. Few computational algorithms and bioinformatics tools are available to interpret the single-cell sequencing data. Low coverage and amplification bias in single-cell sequencing results in allelic dropouts and CNV artifacts. Advances in single-cell sequencing techniques, development of new protocols and software, and integration of single-cell sequencing methods will improve our understanding of cancer biology. Multiomics approach involving parallel investigation of genomic, transcriptomic, epigenomic, and proteomic profiles will help to map the regulations and function of genes at each molecular level. Mass spectrometry-based single-cell proteomics has not gained attention; however, advances in techniques to employ proteomics at single-cell level will certainly help to identify novel diagnostic and prognostic biomarkers, and therapeutic targets in cancer.

References

1. Ren X, Kang B, Zhang Z (2018) Understanding tumor ecosystems by single-cell sequencing: promises and limitations. *Genome Biol* 19(1):211
2. Chen F et al (2015) New horizons in tumor microenvironment biology: challenges and opportunities. *BMC Med* 13:45
3. Yuan Y et al (2016) Role of the tumor microenvironment in tumor progression and the clinical applications (Review). *Oncol Rep* 35(5):2499–2515
4. Guan J, Chen J (2013) Mesenchymal stem cells in the tumor microenvironment. *Biomed Rep* 1(4):517–521
5. Kamdje AHN et al (2017) Mesenchymal stromal cells' role in tumor microenvironment: involvement of signaling pathways. *Cancer Biol Med* 14(2):129–141
6. Arena S et al (2018) Characterization of tumor-derived mesenchymal stem cells potentially differentiating into cancer-associated fibroblasts in lung cancer. *Clin Transl Oncol* 20(12):1582–1591
7. Ishihara S, Ponik SM, Haga H (2017) Mesenchymal stem cells in breast cancer: response to chemical and mechanical stimuli. *Oncoscience* 4(11–12):158–159

8. Fouad YA, Aanei C (2017) Revisiting the hallmarks of cancer. *Am J Cancer Res* 7(5):1016–1036
9. Hanahan D, Weinberg RA (2011) Hallmarks of cancer: the next generation. *Cell* 144(5):646–674
10. Wang M et al (2017) Role of tumor microenvironment in tumorigenesis. *J Cancer* 8(5):761–773
11. Thunnissen E, van der Oord K, den Bakker M (2014) Prognostic and predictive biomarkers in lung cancer. A review. *Virchows Arch* 464(3):347–358
12. McNamara MG, Sahebjam S (2013) Mason WP (2013) Emerging biomarkers in glioblastoma. *Cancers (Basel)* 5(3):1103–1109
13. Henry NL, Hayes DF (2012) Cancer biomarkers. *Mol Oncol* 6(2):140–146
14. Roma-Rodrigues C et al (2019) Targeting tumor microenvironment for cancer therapy. *Int J Mol Sci* 20(4):840
15. Kim J, Bae JS (2016) Tumor-associated macrophages and neutrophils in tumor microenvironment. *Mediat Inflamm* 2016:6058147
16. Quail DF, Joyce JA (2013) Microenvironmental regulation of tumor progression and metastasis. *Nat Med* 19(11):1423–1437
17. Hida K et al (2018) Contribution of tumor endothelial cells in cancer progression. *Int J Mol Sci* 19(5):1272
18. Maishi N, Hida K (2017) Tumor endothelial cells accelerate tumor metastasis. *Cancer Sci* 108(10):1921–1926
19. Onimaru M, Yonemitsu Y (2011) Angiogenic and lymphangiogenic cascades in the tumor microenvironment. *Front Biosci (Schol Ed)* 3:216–225
20. Hui L, Chen Y (2015) Tumor microenvironment: sanctuary of the devil. *Cancer Lett* 368(1):7–13
21. Wu T, Dai Y (2017) Tumor microenvironment and therapeutic response. *Cancer Lett* 387:61–68
22. Walker C, Mojares E, Del Rio Hernandez A (2018) Role of extracellular matrix in development and cancer progression. *Int J Mol Sci* 19(10):3028
23. Sokolenko AP, Imyanitov EN (2018) Molecular diagnostics in clinical oncology. *Front Mol Biosci* 5:76
24. Ryu D et al (2016) Deciphering intratumor heterogeneity using cancer genome analysis. *Hum Genet* 135(6):635–642
25. Wei Q et al (2017) Multiregion whole-exome sequencing of matched primary and metastatic tumors revealed genomic heterogeneity and suggested polyclonal seeding in colorectal cancer metastasis. *Ann Oncol* 28(9):2135–2141
26. Varon-Gonzalez C, Navarro N (2019) Epistasis regulates the developmental stability of the mouse craniofacial shape. *Heredity (Edinb)* 122(5):501–512
27. Racle J et al (2017) Simultaneous enumeration of cancer and immune cell types from bulk tumor gene expression data. *Elife* 6:e26476
28. Lo PK, Zhou Q (2018) Emerging techniques in single-cell epigenomics and their applications to cancer research. *J Clin Genom* 1(1)
29. Bartoschek M et al (2018) Spatially and functionally distinct subclasses of breast cancer-associated fibroblasts revealed by single cell RNA sequencing. *Nat Commun* 9(1):5150
30. Li H et al (2018) Author correction: reference component analysis of single-cell transcriptomes elucidates cellular heterogeneity in human colorectal tumors. *Nat Genet* 50(12):1754
31. Muller S et al (2017) Single-cell profiling of human gliomas reveals macrophage ontogeny as a basis for regional differences in macrophage activation in the tumor microenvironment. *Genome Biol* 18(1):234
32. Valdes-Mora F et al (2018) Single-cell transcriptomics in cancer immunobiology: the future of precision oncology. *Front Immunol* 9:2582
33. Navin N et al (2011) Tumour evolution inferred by single-cell sequencing. *Nature* 472(7341):90–94

34. Wang Y et al (2014) Clonal evolution in breast cancer revealed by single nucleus genome sequencing. *Nature* 512(7513):155–160
35. Francis JM et al (2014) EGFR variant heterogeneity in glioblastoma resolved through single-nucleus sequencing. *Cancer Discov* 4(8):956–971
36. Xu X et al (2012) Single-cell exome sequencing reveals single-nucleotide mutation characteristics of a kidney tumor. *Cell* 148(5):886–895
37. Yu C et al (2014) Discovery of biclonal origin and a novel oncogene SLC12A5 in colon cancer by single-cell sequencing. *Cell Res* 24(6):701–712
38. Wu H et al (2017) Evolution and heterogeneity of non-hereditary colorectal cancer revealed by single-cell exome sequencing. *Oncogene* 36(20):2857–2867
39. Li Y et al (2012) Single-cell sequencing analysis characterizes common and cell-lineage-specific mutations in a muscle-invasive bladder cancer. *Gigascience* 1(1):12
40. Gawad C, Koh W, Quake SR (2014) Dissecting the clonal origins of childhood acute lymphoblastic leukemia by single-cell genomics. *Proc Natl Acad Sci USA* 111(50):17947–17952
41. Li C et al (2017) Single-cell exome sequencing identifies mutations in KCP, LOC440040, and LOC440563 as drivers in renal cell carcinoma stem cells. *Cell Res* 27(4):590–593
42. Yang Z et al (2017) Single-cell sequencing reveals variants in ARID1A, GPRC5A and MLL2 driving self-renewal of human bladder cancer stem cells. *Eur Urol* 71(1):8–12
43. Caswell DR, Swanton C (2017) The role of tumour heterogeneity and clonal cooperativity in metastasis, immune evasion and clinical outcome. *BMC Med* 15(1):133
44. Leung ML et al (2017) Single-cell DNA sequencing reveals a late-dissemination model in metastatic colorectal cancer. *Genome Res* 27(8):1287–1299
45. Eirew P et al (2015) Dynamics of genomic clones in breast cancer patient xenografts at single-cell resolution. *Nature* 518(7539):422–426
46. Baxter E et al (2014) Epigenetic regulation in cancer progression. *Cell Biosci* 4:45
47. Chatterjee A, Rodger EJ, Eccles MR (2018) Epigenetic drivers of tumourigenesis and cancer metastasis. *Semin Cancer Biol* 51:149–159
48. Xi Y et al (2018) Histone modification profiling in breast cancer cell lines highlights commonalities and differences among subtypes. *BMC Genom* 19(1):150
49. Li LC, Carroll PR, Dahiya R (2005) Epigenetic changes in prostate cancer: implication for diagnosis and treatment. *J Natl Cancer Inst* 97(2):103–115
50. Kanwal R, Gupta S (2010) Epigenetics and cancer. *J Appl Physiol* 109(2):598–605
51. Litzénburger UM et al (2017) Single-cell epigenomic variability reveals functional cancer heterogeneity. *Genome Biol* 18(1):15
52. Guo H et al (2013) Single-cell methylome landscapes of mouse embryonic stem cells and early embryos analyzed using reduced representation bisulfite sequencing. *Genome Res* 23(12):2126–2135
53. Gaiti F et al (2019) Epigenetic evolution and lineage histories of chronic lymphocytic leukaemia. *Nature* 569(7757):576–580
54. Farlik M et al (2015) Single-cell DNA methylome sequencing and bioinformatic inference of epigenomic cell-state dynamics. *Cell Rep* 10(8):1386–1397
55. Buenrostro JD et al (2015) Single-cell chromatin accessibility reveals principles of regulatory variation. *Nature* 523(7561):486–490
56. Patel AP et al (2014) Single-cell RNA-seq highlights intratumoral heterogeneity in primary glioblastoma. *Science* 344(6190):1396–1401
57. Tirosh I et al (2016) Dissecting the multicellular ecosystem of metastatic melanoma by single-cell RNA-seq. *Science* 352(6282):189–196
58. Giustacchini A et al (2017) Single-cell transcriptomics uncovers distinct molecular signatures of stem cells in chronic myeloid leukemia. *Nat Med* 23(6):692–702
59. Lawson DA et al (2015) Single-cell analysis reveals a stem-cell program in human metastatic breast cancer cells. *Nature* 526(7571):131–135
60. Liu Y, Cao X (2016) Immunosuppressive cells in tumor immune escape and metastasis. *J Mol Med (Berl)* 94(5):509–522

61. Zheng C et al (2017) Landscape of infiltrating T cells in liver cancer revealed by single-cell sequencing. *Cell* 169(7):1342–1356
62. Chung W et al (2017) Single-cell RNA-seq enables comprehensive tumour and immune cell profiling in primary breast cancer. *Nat Commun* 8:15081
63. Roth A et al (2014) PyClone: statistical inference of clonal population structure in cancer. *Nat Methods* 11(4):396–398
64. Miller CA et al (2014) SciClone: inferring clonal architecture and tracking the spatial and temporal patterns of tumor evolution. *PLoS Comput Biol* 10(8):e1003665
65. Zou M, Jin R, Au KF (2018) Revealing tumor heterogeneity of breast cancer by utilizing the linkage between somatic and germline mutations. *Brief Bioinform*
66. Zhang J et al (2014) Intratumor heterogeneity in localized lung adenocarcinomas delineated by multiregion sequencing. *Science* 346(6206):256–259
67. Zhang LL et al (2017) Multiregion sequencing reveals the intratumor heterogeneity of driver mutations in TP53-driven non-small cell lung cancer. *Int J Cancer* 140(1):103–108
68. Ledgerwood LG et al (2016) The degree of intratumor mutational heterogeneity varies by primary tumor sub-site. *Oncotarget* 7(19):27185–27198
69. Yan T et al (2019) Multi-region sequencing unveils novel actionable targets and spatial heterogeneity in esophageal squamous cell carcinoma. *Nat Commun* 10(1):1670
70. Hao JJ et al (2016) Spatial intratumoral heterogeneity and temporal clonal evolution in esophageal squamous cell carcinoma. *Nat Genet* 48(12):1500–1507
71. Kim TM et al (2015) Subclonal genomic architectures of primary and metastatic colorectal cancer based on intratumoral genetic heterogeneity. *Clin Cancer Res* 21(19):4461–4472
72. Ahmed M, Li LC (2013) Adaptation and clonal selection models of castration-resistant prostate cancer: current perspective. *Int J Urol* 20(4):362–371
73. Horning AM et al (2018) Single-cell RNA-seq reveals a subpopulation of prostate cancer cells with enhanced cell-cycle-related transcription and attenuated androgen response. *Cancer Res* 78(4):853–864
74. Casasent AK, Edgerton M, Navin NE (2017) Genome evolution in ductal carcinoma in situ: invasion of the clones. *J Pathol* 241(2):208–218
75. Casasent AK et al (2018) Multiclonal invasion in breast tumors identified by topographic single cell sequencing. *Cell* 172(1–2):205–217
76. Sottoriva A et al (2015) A big bang model of human colorectal tumor growth. *Nat Genet* 47(3):209–216
77. Sarkar S et al (2013) Cancer development, progression, and therapy: an epigenetic overview. *Int J Mol Sci* 14(10):21087–21113
78. Byler S et al (2014) Genetic and epigenetic aspects of breast cancer progression and therapy. *Anticancer Res* 34(3):1071–1077
79. Byler S, Sarkar S (2014) Do epigenetic drug treatments hold the key to killing cancer progenitor cells? *Epigenomics* 6(2):161–165
80. Campbell PJ et al (2010) The patterns and dynamics of genomic instability in metastatic pancreatic cancer. *Nature* 467(7319):1109–1113
81. Sosa Iglesias V et al (2018) Drug resistance in non-small cell lung cancer: a potential for NOTCH targeting? *Front Oncol* 8:267
82. Lopez-Verdin S et al (2018) Molecular markers of anticancer drug resistance in head and neck squamous cell carcinoma: a literature review. *Cancers (Basel)* 10(10):376
83. Ansell A et al (2016) Epidermal growth factor is a potential biomarker for poor cetuximab response in tongue cancer cells. *J Oral Pathol Med* 45(1):9–16
84. Sauna ZE, Ambudkar SV (2001) Characterization of the catalytic cycle of ATP hydrolysis by human P-glycoprotein. The two ATP hydrolysis events in a single catalytic cycle are kinetically similar but affect different functional outcomes. *J Biol Chem* 276(15):11653–11661
85. Hilgendorf C et al (2007) Expression of thirty-six drug transporter genes in human intestine, liver, kidney, and organotypic cell lines. *Drug Metab Dispos* 35(8):1333–1340
86. Haber M et al (2006) Association of high-level MRPI expression with poor clinical outcome in a large prospective study of primary neuroblastoma. *J Clin Oncol* 24(10):1546–1553

87. Friedrich RE, Punke C, Reymann A (2004) Expression of multi-drug resistance genes (mdr1, mrp1, bcrp) in primary oral squamous cell carcinoma. *Vivo* 18(2):133–147
88. Rivlin N et al (2011) Mutations in the p53 Tumor Suppressor Gene: Important Milestones at the Various Steps of Tumorigenesis. *Genes Cancer* 2(4):466–474
89. Holohan C et al (2013) Cancer drug resistance: an evolving paradigm. *Nat Rev Cancer* 13(10):714–726
90. Kobayashi S et al (2005) EGFR mutation and resistance of non-small-cell lung cancer to gefitinib. *N Engl J Med* 352(8):786–792
91. Razis E et al (2011) Evaluation of the association of PIK3CA mutations and PTEN loss with efficacy of trastuzumab therapy in metastatic breast cancer. *Breast Cancer Res Treat* 128(2):447–456
92. Cook KL et al (2014) Hydroxychloroquine inhibits autophagy to potentiate antiestrogen responsiveness in ER+breast cancer. *Clin Cancer Res* 20(12):3222–3232
93. Curtin NJ (2012) DNA repair dysregulation from cancer driver to therapeutic target. *Nat Rev Cancer* 12(12):801–817
94. Nogueira GAS et al (2018) Polymorphisms in DNA mismatch repair pathway genes predict toxicity and response to cisplatin chemoradiation in head and neck squamous cell carcinoma patients. *Oncotarget* 9(51):29538–29547
95. Chae YK et al (2018) Epithelial-mesenchymal transition (EMT) signature is inversely associated with T-cell infiltration in non-small cell lung cancer (NSCLC). *Sci Rep* 8(1):2918
96. Bearzatto A et al (2000) Epigenetic regulation of the MGMT and hMSH6 DNA repair genes in cells resistant to methylating agents. *Cancer Res* 60(12):3262–3270
97. Navin N et al (2010) Inferring tumor progression from genomic heterogeneity. *Genome Res* 20(1):68–80
98. McGranahan N, Swanton C (2017) Clonal Heterogeneity and Tumor Evolution: Past, Present, and the Future. *Cell* 168(4):613–628
99. Van Allen EM et al (2015) Genomic correlates of response to CTLA-4 blockade in metastatic melanoma. *Science* 350(6257):207–211
100. Rizvi NA et al (2015) Cancer immunology. Mutational landscape determines sensitivity to PD-1 blockade in non-small cell lung cancer. *Science* 348(6230):124–128
101. Fares CM et al (2019) Mechanisms of resistance to immune checkpoint blockade: why does checkpoint inhibitor immunotherapy not work for all patients? *Am Soc Clin Oncol Educ Book* 39:147–164
102. Tang F et al (2009) mRNA-Seq whole-transcriptome analysis of a single cell. *Nat Methods* 6(5):377–382
103. Zheng GX et al (2016) Haplotyping germline and cancer genomes with high-throughput linked-read sequencing. *Nat Biotechnol* 34(3):303–311
104. Gierahn TM et al (2017) Seq-Well: portable, low-cost RNA sequencing of single cells at high throughput. *Nat Methods* 14(4):395–398
105. Wang Y, Navin NE (2015) Advances and applications of single-cell sequencing technologies. *Mol Cell* 58(4):598–609
106. Hou Y et al (2016) Single-cell triple omics sequencing reveals genetic, epigenetic, and transcriptomic heterogeneity in hepatocellular carcinomas. *Cell Res* 26(3):304–319
107. Lambrechts D et al (2018) Phenotype molding of stromal cells in the lung tumor microenvironment. *Nat Med* 24(8):1277–1289
108. Ma KY et al (2019) Single-cell RNA sequencing of lung adenocarcinoma reveals heterogeneity of immune response-related genes. *JCI Insight* 4(4):e121387
109. Pantel K, Speicher MR (2016) The biology of circulating tumor cells. *Oncogene* 35(10):1216–1224
110. Zhu Z et al (2018) Progress and challenges of sequencing and analyzing circulating tumor cells. *Cell Biol Toxicol* 34(5):405–415
111. Yadavalli S et al (2017) Data-driven discovery of extravasation pathway in circulating tumor cells. *Sci Rep* 7:43710

112. Krebs MG et al (2014) Molecular analysis of circulating tumour cells-biology and biomarkers. *Nat Rev Clin Oncol* 11(3):129–144
113. Arya SK, Lim B, Rahman AR (2013) Enrichment, detection and clinical significance of circulating tumor cells. *Lab Chip* 13(11):1995–2027
114. Khoob BL et al (2015) Short-term expansion of breast circulating cancer cells predicts response to anti-cancer therapy. *Oncotarget* 6(17):15578–15593
115. Cristofanilli M et al (2004) Circulating tumor cells, disease progression, and survival in metastatic breast cancer. *N Engl J Med* 351(8):781–791
116. Balakrishnan A et al (2019) Circulating Tumor Cell cluster phenotype allows monitoring response to treatment and predicts survival. *Sci Rep* 9(1):7933
117. Lianidou ES, Markou A, Strati A (2015) The role of CTCs as tumor biomarkers. *Adv Exp Med Biol* 867:341–367
118. Shaw JA et al (2017) Mutation analysis of cell-free DNA and single circulating tumor cells in metastatic breast cancer patients with high circulating tumor cell counts. *Clin Cancer Res* 23(1):88–96
119. De Luca F et al (2016) Mutational analysis of single circulating tumor cells by next generation sequencing in metastatic breast cancer. *Oncotarget* 7(18):26107–26119
120. Gao Y et al (2017) Single-cell sequencing deciphers a convergent evolution of copy number alterations from primary to circulating tumor cells. *Genome Res* 27(8):1312–1322
121. Ni X et al (2013) Reproducible copy number variation patterns among single circulating tumor cells of lung cancer patients. *Proc Natl Acad Sci USA* 110(52):21083–21088
122. Carter L et al (2017) Molecular analysis of circulating tumor cells identifies distinct copy-number profiles in patients with chemosensitive and chemorefractory small-cell lung cancer. *Nat Med* 23(1):114–119
123. Yu M et al (2013) Circulating breast tumor cells exhibit dynamic changes in epithelial and mesenchymal composition. *Science* 339(6119):580–584
124. Ting DT et al (2014) Single-cell RNA sequencing identifies extracellular matrix gene expression by pancreatic circulating tumor cells. *Cell Rep* 8(6):1905–1918
125. Miyamoto DT et al (2015) RNA-Seq of single prostate CTCs implicates noncanonical Wnt signaling in antiandrogen resistance. *Science* 349(6254):1351–1356
126. Grillet F et al (2017) Circulating tumour cells from patients with colorectal cancer have cancer stem cell hallmarks in ex vivo culture. *Gut* 66(10):1802–1810
127. Baslan T et al (2012) Genome-wide copy number analysis of single cells. *Nat Protoc* 7(6):1024–1041
128. Leung ML et al (2015) SNES: single nucleus exome sequencing. *Genome Biol* 16:55
129. Gao R et al (2017) Nanogrid single-nucleus RNA sequencing reveals phenotypic diversity in breast cancer. *Nat Commun* 8(1):228
130. Macosko EZ et al (2015) Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. *Cell* 161(5):1202–1214
131. Grindberg RV et al (2013) RNA-sequencing from single nuclei. *Proc Natl Acad Sci USA* 110(49):19802–19807
132. Shapiro E, Biezuner T, Linnarsson S (2013) Single-cell sequencing-based technologies will revolutionize whole-organism science. *Nat Rev Genet* 14(9):618–630
133. Bernard V et al (2019) Single-cell transcriptomics of pancreatic cancer precursors demonstrates epithelial and microenvironmental heterogeneity as an early event in neoplastic progression. *Clin Cancer Res* 25(7):2194–2205
134. Kim C et al (2018) Chemoresistance evolution in triple-negative breast cancer delineated by single-cell sequencing. *Cell* 173(4):879–893
135. Azizi E et al (2018) Single-cell map of diverse immune phenotypes in the breast tumor microenvironment. *Cell* 174(5):1293–1308
136. Liu M et al (2017) Multi-region and single-cell sequencing reveal variable genomic heterogeneity in rectal cancer. *BMC Cancer* 17(1):787
137. Lavin Y et al (2017) Innate immune landscape in early lung adenocarcinoma by paired single-cell analyses. *Cell* 169(4):750–765

138. Dey SS et al (2015) Integrated genome and transcriptome sequencing of the same cell. *Nat Biotechnol* 33(3):285–289
139. Wu L et al (2015) Full-length single-cell RNA-seq applied to a viral human cancer: applications to HPV expression and splicing analysis in HeLa S3 cells. *Gigascience* 4:51
140. Dago AE et al (2014) Rapid phenotypic and genomic change in response to therapeutic pressure in prostate cancer inferred by high content analysis of single circulating tumor cells. *PLoS ONE* 9(8):e101777
141. Zhao L et al (2013) High-purity prostate circulating tumor cell isolation by a polymer nanofiber-embedded microchip for whole exome sequencing. *Adv Mater* 25(21):2897–2902