MONASH University

Minimum Message Length Inference of Protein Alignments and Statistical Models of Amino Acid Evolution

by

Dinithi Navodhya Sumanaweera

A thesis submitted for the degree of Doctor of Philosophy at

Monash University in 2021

Faculty of Information Technology

Minimum Message Length Inference of Protein Alignments and Statistical Models of Amino Acid Evolution

Faculty of Information Technology
Monash University
Australia

This thesis was typeset with LaTeX 2$_\varepsilon$ by the author. LaTeX 2$_\varepsilon$ is an extension of LaTeX. LaTeX is a collection of macros for TeX. TeX is a trademark of the American Mathematical Society. The document class used in formatting this thesis was written by Glenn Maughan and modified by Dean Thompson and David Squire of Monash University.

"I was taught that the way of progress was neither swift nor easy"

– Marie Skłodowska Curie (1867-1934)

Contents

Minimum Message Length Inference of Protein Alignments and Statistical Models of Amino Acid Evolution

Dinithi Navodhya Sumanaweera
Monash University, 2021

Supervisors:
Dr. Arun S. Konagurthu & Dr. Lloyd Allison

Abstract

Proteins are biological macromolecules responsible for a panoply of critical functions within the cells of living organisms. Protein comparison via the technique of alignment is an indispensable first step in many biological studies. An alignment postulates a one-to-one correspondence between pairs of amino acids amongst proteins, reflecting how they are evolutionarily related. This relationship can be inferred with varying degrees of confidence, using information from either the one-dimensional amino acid sequence or the three-dimensional structure, and sometimes both.

The classical problem of *protein sequence alignment* has been a well investigated topic for the last four decades. A general sequence alignment method involves an objective function to quantify the quality of an alignment relationship (in terms of amino acid substitutions and gaps) and an optimisation algorithm to search for the best alignment. Despite a lot of attention, generating sequence alignments in practice relies on *ad hoc* scoring functions and user-specified parameter choices, yielding radically different and contradictory alignments under varying parameter settings. Moreover, there is a disconnect between the mathematical models that handle amino acid substitutions and those that handle gaps. Furthermore, the trade-off between the complexity of an alignment and its fidelity to explain the underlying sequence data has received little attention.

This thesis is an attempt to revisit this classical problem in the hope of addressing the aforementioned limitations. It approaches the pairwise protein sequence alignment problem from the standpoint of *unsupervised inductive inference* using the Bayesian and information-theoretic criterion of *Minimum Message Length* (MML). The thesis develops a statistical framework to generate and evaluate alignment relationships over sequence data based on their joint or marginal probabilities, and does so strictly in terms of Shannon Information, measurable in bits. The MML objective provides a direct way to handle the trade-off between alignment/hypothesis complexity and fit, and to explore competing alignments beyond finding the best under the objective. The framework further provides the ability to learn statistical models of alignments in the form of three-state machine models, parameterised on the (evolutionary) time of divergence, to work in conjunction with a given time-dependent model of amino acid substitution. This thesis culminates with the inference of a new Markov model of amino acid substitution (named `MMLSUM`). A comprehensive study of `MMLSUM` compared to popularly used substitution models has been carried out to demonstrate its effectiveness. Finally, the thesis concludes by proposing a proof-of-concept approach to combine 1D sequence and 3D structure information under a single unified MML framework, by combining the models for protein sequence alignment presented in this thesis with existing (in-house) models for protein structure alignment.

Minimum Message Length Inference of Protein Alignments and Statistical Models of Amino Acid Evolution

Declaration

This thesis is an original work of my research and contains no material which has been accepted for the award of any other degree or diploma at any university or equivalent institution and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

Dinithi Navodhya Sumanaweera
March 3, 2021

Publications & Software

Below lists the scientific manuscripts and the software material arising from this thesis.

Manuscripts in communication:

- **Dinithi Sumanaweera**, Lloyd Allison and Arun S. Konagurthu, Bridging the Gaps in Statistical Models of Protein Alignment. ***In communication.***
  arχiv preprint: https://arxiv.org/abs/2010.00855

Manuscripts published during the candidature:

- **Dinithi Sumanaweera**, Lloyd Allison and Arun S. Konagurthu (2018), The bits between proteins. In proceedings of the ***Twenty-Eighth IEEE Data Compression Conference (DCC 2018)***, pp. 177-186.
  DOI: https://doi.org/10.1109/DCC.2018.00026

- **Dinithi Sumanaweera**, Lloyd Allison and Arun S. Konagurthu (2019), Statistical compression of protein sequences and inference of marginal probability landscapes over competing alignments using finite state models and Dirichlet priors. ***Bioinformatics***, **35**(14):i360-i369.
  DOI: https://doi.org/10.1093/bioinformatics/btz368

  – Presented in the proceedings track of the ***Twenty-Seventh annual international conference on Intelligent Systems for Molecular Biology*** and the ***Eighteenth European Conference on Computational Biology (ISMB/ECCB 2019)***, July 21-25 2019, Basel, Switzerland.

Software and Material:

- **SeqMMLigner:** Minimum Message Length based aligner of protein (amino acid) sequences
  Available at: https://lcb.infotech.monash.edu/seqmmligner

- **MMLSUM:** Minimum Message Length amino acid SUbstitution Matrix (modelling evolutionary amino acid interchanges in protein sequence)
  Available at: https://lcb.infotech.monash.edu.au/mmlsum

List of Commonly Used Abbreviations

1D	One-Dimensional
3D	Three-Dimensional
DP	Dynamic Programming
EM	Expectation-Maximisation
GO	Gene Ontology
KL	Kullback-Leibler
ML	Maximum Likelihood
DNA	Deoxyribonucleic Acid
DPA	Dynamic Programming Algorithm
HMM	Hidden Markov Model
MML	Minimum Message Length
PAM	Point Accepted Mutation
PCA	Principal Component Analysis
PDB	Protein Data Bank
PDF	Probability Density Function
PMF	Probability Mass Function
RNA	Ribonucleic Acid
CATH	Class, Architecture, Topology, Homologous superfamily
CTMC	Continuous Time Markov Chain
DTMC	Discrete Time Markov Chain
RMSD	Root-mean-square deviation
SCOP	Structural Classification of Proteins
tSNE	t-distributed Stochastic Neighbour Embedding
PDB	Protein Data Bank
BLOSUM	BLOcks SUbstitution Matrix
MMLSUM	Minimum Message Length SUbstitution Matrix
PFASUM	PFAm SUbstitution Matrix
SABMARK	Sequence Alignment BenchMARK
UniProt	Universal Protein Resource
HOMSTRAD	HOMologous STRucture Alignment Database

List of Tables

List of Figures

Acknowledgements

It was indeed a splendid journey with many bright peaks and challenging valleys to conquer along the way. I would like to express my heartiest gratitude to all the people who have made it possible.

First and foremost, I am forever grateful to my wonderful supervisors, Dr. Arun Konagurthu and Dr. Lloyd Allison, for their immense support, guidance, advice and encouragement. From the very beginning, I knew I was in the presence of greatness and conscientiousness. They have always inspired and motivated me. Regular meetings and discussions with them were always a delight; those were filled with intellectual gravity and enthusiasm, not only revolving around the main research topic but also on currently important themes in the field and various philosophies. I am thankful that they recognised and understood me for who I am with my strengths and weaknesses, and constantly helped me in every possible way to develop and improve myself.

Arun is always cheerful, spreading merriment and optimism. Through our daily interactions, he showed me the path to build a solid foundation of technical, analytical and research skills. Whenever I came across a complex theory or concept, he persuaded me to work it out from the first principles, teaching me how to understand the intuition behind. Arun has a unique and impressive way of approaching problems with his meticulous research practices and a deep understanding of diverse subjects, which was pivotal to my endeavour in realising this thesis.

Lloyd is truly a master, with a serene and radiant presence. His attention to detail and the way he put complex ideas into simpler and concise terms is remarkable. It is always a joy to listen to him. There is so much to learn from his decades of experience in Computer Science and Bioinformatics. His insights, views and feedback throughout my candidature were indispensable and extremely helpful in ensuring that this thesis was heading in the right direction.

I am extremely lucky to have met these two brilliant mentors and worked under their wisdom and kindness. Thank you very much for giving me the opportunity and for being patient with me all this time while I was learning.

Next, my sincere thanks goes to all my milestone panel members: Dr. Francois Petitjean, Assoc. Prof. Gholamreza Haffari and chairpersons: Dr. Ron Steinfeld, Dr. Mario Boley, for their constructive feedback and support. I extend my sincere appreciation to Rekha Amar, for her words of encouragement, care and well wishes. Special thanks goes to James Collier for his kind support and being there with Arun to cheer at my conference presentation in Basel. I am also thankful to Assoc. Prof. Sureshkumar Balasubramanian for his best wishes.

Further I wish to express my deepest gratitude to the past and present staff of the graduate research office and the operations team at the Monash Faculty of IT (FIT). Especially, Danette Deriane; I still remember the warm welcome I received from you on my very first day. Thank you for being a good friend to all PhD students. Many thanks to Julie Holden, Helen Cridland and Rachael Unwin for your kind support.

I also extend my sincere thanks to Monash University for awarding me with the Monash Graduate Scholarship (MGS) and Monash International Postgraduate Research Scholarship

(MIPRS), and the Australian Government for granting me the opportunity to study in Australia. Moreover, I wholeheartedly appreciate the generous support provided by the FIT, Monash Graduate Research Office, Australian Research Council Discovery Project and the EMBL-Australia for conference and workshop travel during my candidature. This course of study greatly benefitted from all infrastructure and resources provided by the FIT and Monash University, including Monash Library Services and Monash Advanced Research Computing Hybrid. Furthermore, I wish to pay my special regards to the Monash Residential Services for providing me with a safe and sound homely stay away from home at Briggs Hall.

This exciting and joyous period would not have been possible without my dear friends. Thank you Bhagya Hettige, Kasun Bandara and Laksri Wijeratne for your tremendous support and those insightful and fun conversations we had all throughout. Thank you Hashini Senaratne and Yasura Vithana for everything. Your enormous support, kindness and care means a lot to me. Thank you Dilini Rajapaksha for all those memorable chats, making the lab atmosphere so enjoyable to work in. My sincere thanks goes to Hansika Hewamalage, Harsha Perera, Sachith Seneviratne, Ruangsak Trakunphutthirak, Ying Yang, Kai Siong Yow, Komal Komal, Mohammad Samiullah, Xuhui Zang, Ehsan Shareghi, Han Duy Phan, Chaluka Salgado, Abida Shahzad, Harald Bogeholz, Malika Rathnayake, Sanduni Rajapakse and many others whom I had the pleasure of getting to know. Whether it was a long or short conversation over coffee, lunch/dinner hangout, meeting in corridors, or working together during various events, thank you all for giving me so many memories to cherish for life. You all have a special place in my heart.

I owe a great debt of gratitude to my dear friend Amanda Gunawardena for always being there for me and listening to me, with emotional support during happy and stressful times. I am also indebted to Aunty Gayanie Ratnayake and family for being just a one call away, looking after me during my stay in Melbourne far away from home.

I would also like to take this opportunity to express my thanks to all my teachers I learned from during my schooling time at Musaeus College, Colombo and University of Moratuwa in Sri Lanka, for showing me the path to knowledge; Especially, Dr. Amal Shehan Perera and Prof. Gihan Dias, for recommending me to Monash for pursuing a PhD.

During every phase and step taken, I was blessed with the unconditional support I receive from my loving mother and father, the light of my life, being with me through every thick and thin, and always believing in me and encouraging me to do my best. Words cannot express how grateful I am. Thank you for instilling me a passion for Science. Also to my beloved grandma, thank you for having faith in me and always uplifting my confidence. I am here today because of them. And my loving brother, thank you for always caring and being there for me. Last but not least, I wish to extend my sincere appreciation to all my close relatives: my grandparents, aunts, uncles and cousins in my family for their constant affection, care and support.

Reflecting back, this journey was no smooth ride, yet a precious phase of many learnings and experiences. It was realised amidst your blessings, inspiration, nurture and well wishes. Each and every one of you has my deepest, most profound and heartfelt gratitude.

Dinithi N. Sumanaweera

Monash University
February 2021

Chapter 1

Introduction

"Begin at the beginning", the King said, very gravely,
"and go on till you come to the end: then stop."

– Lewis Carroll in Alice in Wonderland

Proteins are biological macromolecules that play crucial roles in the molecular and cellular processes of all living organisms. They carry out a myriad of functions within cells, as antibodies, transporters, enzymes, regulators, and signal transducers, amongst many others (Lesk, 2016). The chemical signature of a protein is characterised by a linear chain of amino acids (*sequence*) that spontaneously folds into an intricate three-dimensional shape (*structure*), which in turn determines its function.

Over the past seven decades, technological developments in molecular biology have fuelled rapidly-growing public databases of protein sequences and their atomic-resolution structures (e.g. Universal Protein Resource – UniProt (UniProt Consortium and others, 2017) and Protein Data Bank – PDB (Berman et al., 2003), respectively).

Comparison between extant proteins is an important task in many biological studies. It reveals the principles and macromolecular consequences of evolution, whose insights drive progress in life sciences and medicine (Rosenberg, 2009).

Protein comparison often relies on the computational task of *alignment*. An *alignment* is the assignment of one-to-one correspondences between a subset of their amino acids. Such an alignment is used to answer at least two fundamental questions: *Are the proteins being compared related? If yes, how precisely are they related, at the level of their amino acid correspondences?* Each correspondence (*match*) between amino acids is a hypothesis of their divergence from the same locus within the genome of a *nonextant* common ancestor (Barton and Sternberg, 1987; Konagurthu et al., 2006). On the other hand, unmatched amino acids are taken as accumulated *insertions* or *deletions* during the evolution of proteins. Thus, when alignments are computed reliably, they reveal how proteins diverge, tolerating changes to their sequence, while conserving core elements of their structure and consequently, their function (Lesk, 2016).

Alignment relationships between proteins are commonly derived using two distinct sources of information: the one-dimensional (1D) *sequence* composed of amino acids symbols, or the three-dimensional (3D) *structure* composed of protein coordinates. Based on the information used to generate an alignment, the outcome is designated as either a *sequence alignment* or a *structure alignment*. When both sources of information are combined, it is designated as a *sequence-structure alignment* (Levitt and Gerstein, 1998).

In general, the protein alignment problem is posed as an optimisation problem of finding the *best* alignment under some objective function. This objective function is used to quantify the *quality* of any alignment. Almost ubiquitously, two key (opposing) criteria are involved

here: (1) a criterion that accounts for *matched* regions, and (2) a criterion to account for the *unmatched* regions. An objective function is *constructed* to balance them. Depending on the type of alignment (i.e. sequence, structure or sequence-structure alignment) being computed, these criteria are formulated in different ways. Based on these applied criteria and the parameter choices being made, it remains that, finding biologically-meaningful relationships between proteins is a challenging computational problem (Sali and Blundell, 1990; Jones et al., 1992b; Bryant and Altschul, 1995; Smith et al., 1997; Levitt and Gerstein, 1998; Rost, 1999; Buchan and Jones, 2017).

From the point of statistical learning, the varied dispensations of these criteria (across different alignment types) in the literature provide handy yet approximate ways to address the trade-off between descriptive *complexity* (of an alignment relationship) and *fit* (fidelity of that relationship). Specifically, in traditional methods for sequence alignment, this trade-off is commonly handled using substitution scores and gap penalties. For structure alignment, this is commonly handled using the alignment coverage (proportion of matches) and the root-mean-square deviation after their rigid-body superposition of protein coordinates.

Comprehensive reviews have shown that, different formulations of the alignment objective function and parameters controlling them have led to a lack of consensus amongst the resultant protein alignments in the existing state of the art. Tuning and searching for parameters that are *best* for each alignment run is a challenging problem and remains a major source of inconsistencies between alignments generated by varying methods (Fitch and Smith, 1983; Barton and Sternberg, 1987; Vingron and Waterman, 1994; Blake and Cohen, 2001; Do et al., 2005, 2006; Hasegawa and Holm, 2009; Slater et al., 2012). Biological studies that rely on inaccurate alignment results lead to erroneous conclusions and interpretations (Löytynoja and Goldman, 2008).

1.1 Motivation and Overarching Methodology

This thesis revisits the classical problem of protein *sequence alignment*, in an attempt to improve its statistical foundations and overcome key limitations observed in commonly-used alignment methods (see §2.4).

Broadly, it approaches the problem of protein sequence alignment from the standpoint of unsupervised *inductive inference*, where each hypothesis is evaluated based on its ability to *explain* itself as well as the observed data, supported by probabilistic models whose parameters are automatically estimated/inferred (rather than tuned by users).

Specifically, this thesis approaches the inference of sequence alignments using the general Bayesian and information-theoretic technique of *Minimum Message Length* (MML) (Wallace and Boulton, 1968; Wallace, 2005). Here, the quality of any alignment relationship between amino acid sequences is quantified in terms of its information-theoretic *complexity* and its *fidelity* to succinctly explain the observed amino acid sequence data. Using MML, this can be formalised as the length of the lossless encoding of any proposed alignment relationship/hypothesis and the amino acid sequence data it explains – i.e. the *Shannon information content* – measurable in *bits*. Such an information-theoretic dispensation of this problem yields a powerful framework with important properties (see §4.1) and provides a formal way to address the trade-off between alignment complexity and its ability to explain the sequence data.

Below, the specific objectives of this thesis are made concrete, followed by key contributions and a chapter-wise outline.

1.2 Objectives of this Thesis

This thesis has the following key objectives:

- Develop a new unsupervised statistical framework for the protein sequence alignment problem by modelling it using the Bayesian and information-theoretic criterion of Minimum Message Length (MML) inference (see §4.1 – §4.2).

- Extend this framework to allow exploration of competing alignments under the criteria of joint and marginal (i.e. total) probabilities of alignments over protein sequences, leading to useful visualisation of the entire alignment landscape (see §4.3).

- Learn time-dependent alignment three-state machine models to work in concert with any specified Markov model of amino acid substitutions, thus statistically unifying the treatment of all amino acid substitutions, insertions and deletions (see Chapter 5).

- Infer new Markov models of amino acid substitution that are versatile across the range of amino acid relationships (see Chapter 6)

- Combine the outcomes from this thesis with existing MML models for protein *structure* alignment, thereby providing a new proof-of-concept information framework to generate protein *sequence-structure* alignment (see Chapter 7).

1.3 Contributions

This thesis contributes to the theory and practice of protein sequence alignment, by reformulating the classical problem using formal methods of Minimum Message Length (MML) inference. The developed framework (see Chapter 4) not only provides the ability to search for a single best alignment, but also supports the exploration of competing alignments and visualisation of the entire *landscape* of all possible alignment relationships.

Furthermore, this thesis presents an inference of a complete set of statistical models quantifying the evolution of protein sequences (see Chapter 5-6). Specifically, the framework infers new ("time"-dependent) stochastic Markov models of amino acid substitution that outperform widely used substitution matrices and their companion (time-dependent) alignment three-state machine models, overcoming a key shortcoming in the field: that of mathematically unifying amino acid substitutions with amino acid insertions and deletions under the same framework.

Finally, beyond developing the MML framework and deriving statistical models for explaining amino acid evolution, this thesis culminates with a proof-of-principle approach to combine sequence and structure information in a unified framework (see Chapter 7). Specifically, the sequence models developed within this thesis is combined with the (in-house) MML-based *structure* models and alignment framework developed recently by the PhD thesis of Collier (2016), in order to address the *sequence-structure alignment* problem.

1.4 Thesis Outline

The rest of the thesis is structured as follows.

Chapter 2 gives the background required to lay out the specifics of the protein sequence alignment problem. It provides a broad overview of proteins, and introduces the common data sources and collections to prompt the material used in benchmarking the improvements presented in this thesis. The chapter also introduces the common methodologies used to construct protein sequence alignments, along with some of their deficiencies that motivate the research in this thesis.

Chapter 3 introduces the methodological background central to the techniques employed in this thesis, covering basics of information theory and statistical inductive inference. Specifically, this chapter discusses the general method of Minimum Message Length (MML) inference, and how it can be used to carry out unsupervised estimation of statistical model parameters. It presents technical details of the methodological building blocks that underpin the key contributions of the ensuing chapters.

Chapter 4 develops the MML framework for protein alignment and explores its statistical properties. It explains the essential statistical models that support this framework for deducing alignment relationships between sequences. Amongst other things, this chapter explores the construction of a *null model* of amino acids, inferred using MML estimation techniques over a set of proteomes across various species. The chapter introduces a model of protein alignment in information-theoretic terms by explaining the formulation and estimation of the *joint probability* between an alignment and the sequences whose relationship it describes. Further, it goes into detailing the estimation of *marginal (total) probability* over all possible alignments, to explore relationships between sequences in unspecified ways. These models follow a test of alignment significance that can be posed naturally in MML formulations, using the notion of *lossless compression* measured against the length of encoding protein sequences under a *null model*. Finally, the concept of an alignment landscape is explored for visualising competing alignments under the above models of relationships between sequences. This chapter covers all material originally described and published in (Sumanaweera et al., 2018) along with some parts published in (Sumanaweera et al., 2019).

Chapter 5 describes an unsupervised inference of the statistical models of amino acid evolution as a function of estimated divergence between two proteins, extending the initial MML framework developed in Chapter 4. Specifically, this chapter focuses on the inference of alignment three-state machines as a function of sequence divergence (evolutionary time) for any stated Markov model of amino acid substitution. It begins by introducing Dirichlet probability distributions to model the free parameters of an alignment three-state machine, and shows the MML method of inferring them. Next, it elaborates on the estimation of time-dependent Dirichlet models. This chapter contains material originally described and published in (Sumanaweera et al., 2019), with results of comparing the MML alignment framework with a set of existing global alignment programs.

Chapter 6 completes the full set of statistical models necessary for addressing the protein sequence alignment problem, building on the previous research chapters 4 and 5. Specifically, it explains the *joint* unsupervised inference of a Markov model of amino acid substitution together with the corresponding inference of Dirichlet models (and thereby the three-state machine parameters – explored in the previous chapter), over any given benchmark containing protein alignments. The chapter carries out a systematic comparison of existing amino acid

substitution models, in terms of the Shannon information content required to describe a wide-variety of alignment benchmarks using those models. It also explores qualitative aspects of various substitution matrices. This chapter contains material presented in (Sumanaweera et al., 2020) (manuscript under review at the time of writing this thesis).

Chapter 7 is the last of the main research chapters which presents a *proof-of-principle* that permits the unification of 1D protein sequence information and 3D structure information under the MML information-theoretic framework, to generate *sequence-structure* alignments of proteins. Specifically, this chapter explores a Naïve Bayes method for integrating the MML-based *sequence alignment* models presented in this thesis with the previously established, MML-based *structure alignment* models introduced by the thesis of Collier (2016). Preliminary results are presented, showing a quantitative comparison of similarities and differences between structure(-only) alignments with those that combine sequence and structure. A discussion using a case study is explored to understand the qualitative differences between these two alignment types.

Chapter 8 concludes the thesis by reflecting on the key outcomes of this research endeavour, and future directions that can build on this work.

Chapter 2

Proteins and their Alignment

"*In the drama of life on a molecular scale, proteins are where the action is*"

– Arthur M. Lesk

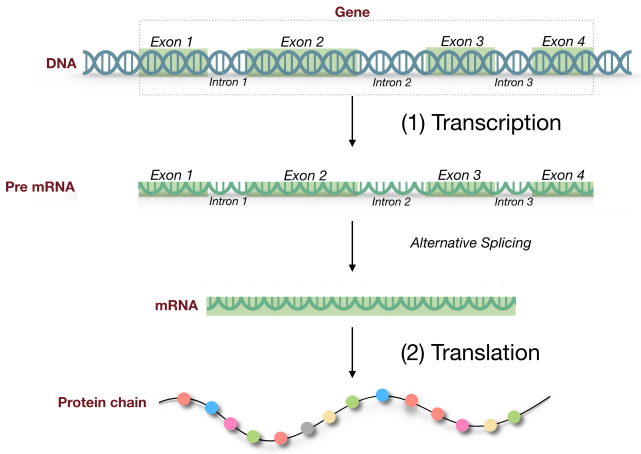
This chapter gives a broad overview of proteins. It covers how proteins are translated within cells, the properties of their constituent amino acids, relevant aspects of how proteins evolve, and the data streams/sources and classification schemes useful for this thesis. It also covers the technical details on the problem of protein sequence alignment with a discussion on the current lacunae in its state of the art. Mainly, all the sections presented here establish the motivation behind the research of this thesis.

2.1 Introduction to Proteins

Deoxyribonucleic acid (DNA) is the genetic blueprint that encodes all life on Earth. Cellular machinery is able to decode this genetic information through a series of transformations resulting in proteins (see Figure 2.1).

The expressed proteins in living organisms control nearly all biological processes from its origin to its development and maintenance (Lesk, 2016). For example, haemoglobin is a protein which is responsible for transporting oxygen; insulin regulates the glucose level in blood; and fibrin facilitates blood clotting. An abnormality or disruption with regards to a protein and its function could impede regular cellular activities and essential biological pathways, causing adverse phenotypes, diseases and even death. Hence, studying and understanding them is of paramount importance, especially for developing therapeutic drugs.

A protein is composed of linear chain(s) of amino acids, where typically, each amino acid comes from a canonical set of twenty naturally-occurring amino acids (See Table 2.1). Chemically, all amino acids are composed of an amine group and a carboxylic acid group, linked via a central (α) carbon atom. However, the amino acids distinguish from each other based on the chemical signature of atoms that are bound to the central carbon, giving the amino acids varying physicochemical characteristics (See Figure 2.2). Successive amino acids along the protein chain are linked via peptide bonds that are formed between the carboxylic carbon and the amino nitrogen atoms. Thus, any protein chain can be represented as a *sequence* of



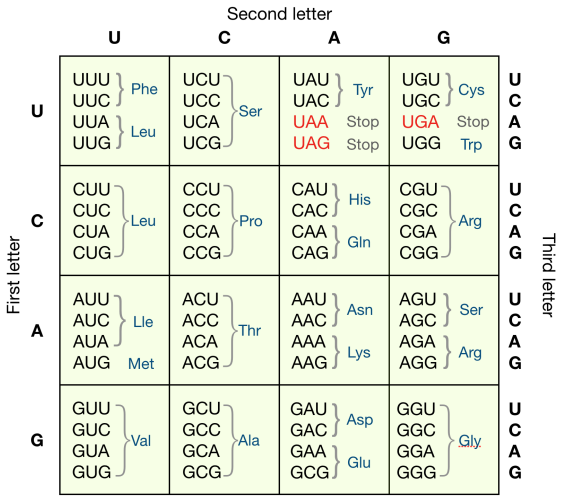


Figure 2.1: The *central dogma of molecular biology:* Genome composed of DNA (described over the {A,T,G,C} alphabet) encodes all instructions required for constructing proteins through genes. Genes are the specific information units within the genome that code for proteins. A gene is transcribed into mRNA (described over the {A,U,G,C} alphabet) through an intermediate Pre-messenger RNA (Pre-mRNA) molecule. Molecular machinery within cells converts each successive group of three bases (*codon*) in the RNA molecule into an amino acid. The mapping of 64 possible codons to their corresponding 20 possible amino acids defines the *universal genetic code*.

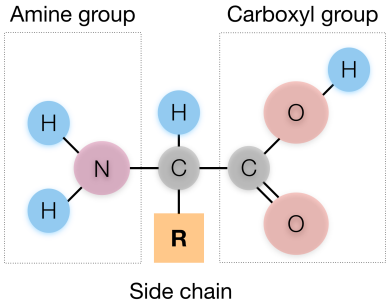


Figure 2.2: Chemical structure of a general amino acid. It has an amine group (NH_2), a carboxylic group (COOH), and a central carbon atom with a side chain (R) attached to it. R determines the type of the amino acid.)

amino acids along its chain. Proteins spontaneously folds into intricate three-dimensional (3D) shapes (*structure*), upon translation from mRNA (see Figure 2.1).[1]

Amino acid properties such as hydrophobicity, polarity, and charge of an amino acid residue determine the types of bonds they form and their interactions with their surroundings. For instance, hydrophobic amino acid are usually buried within the interior of the protein structure (Lesk, 2010).

[1] Levinthal (1969) reasoned that the minimum free energy state cannot feasibly be a result of an exhaustive search of the astronomically-large space of potential protein shapes, thereby hinting that the optimised shape of the protein is kinetically-driven.

Table 2.1: This is the list of 20 naturally occurring amino acid residues. Each residue is identified by a unique name, with a three-letter abbreviation and a corresponding single-letter code (IUPAC-IUB Committee on Biochemistry Nomenclature, 1968).

Amino acid	Three-letter code	Single-letter code	Formula	Chemical class*
Alanine	Ala	A	$C_3H_7NO_2$	aliphatic
Arginine	Arg	R	$C_6H_14N_4O_2$	basic
Asparagine	Asn	N	$C_4H_8N_2O_3$	amide
Aspartic acid	Asp	D	$C_4H_7NO_4$	acidic
Cysteine	Cys	C	$C_3H_7NO_2S$	sulfur
Glutamine	Gln	Q	$C_5H_10N_2O_3$	amide
Glutamic acid	Glu	E	$C_5H_9NO_4$	acidic
Glycine	Gly	G	$C_2H_5NO_2$	aliphatic
Histidine	His	H	$C_6H_9N_3O_2$	basic
Isoleucine	Ile	I	$C_6H_{13}NO_2$	aliphatic
Leucine	Leu	L	$C_6H_{13}NO_2$	aliphatic
Lysine	Lys	K	$C_6H_{14}N_2O_2$	basic
Methionine	Met	M	$C_5H_{11}NO_2S$	sulfur
Phenylalanine	Phe	F	$C_9H_{11}NO_2$	aromatic
Proline	Pro	P	$C_5H_9NO_2$	aliphatic
Serine	Ser	S	$C_3H_7NO_3$	hydroxyl
Threonine	Thr	T	$C_4H_9NO_3$	hydroxyl
Tryptophan	Trp	W	$C_{11}H_{12}N_2O_2$	aromatic
Tyrosine	Tyr	Y	$C_9H_{11}NO_3$	aromatic
Valine	Val	V	$C_5H_{11}NO_2$	aliphatic

*Chemical class of each amino acid is given by the IMGT classification (Lefranc et al., 2015).

2.2 Organisation of Proteins

Proteins are commonly rationalised using the following levels of description (Linderstrøm-Lang, 1952) (see Figure 2.3).

Primary structure denotes the linear chain of amino acids in a protein, represented as a one-dimensional string of characters, often from the canonical 20-letter amino acid alphabet.

Secondary structure describes the local periodic substructural patterns, helices and strands-of-sheets (Pauling et al., 1951; Pauling and Corey, 1951)

Tertiary structure refers to the overall three dimensional conformation a protein chain attains by folding into a stable 3D shape. The position of each atom in 3D space is denoted by its (x, y, z) coordinates.

Quaternary structure defines an ensemble of tertiary structures forming a larger complex. This level of description potentially integrates multiple protein chains which interact cohesively as subunits

In addition to the aforementioned broad descriptions of the protein structure, proteins are also commonly described using their *domains*. A domain is a compact, independently evolving structural unit (Richardson, 1981; Murzin et al., 1995)

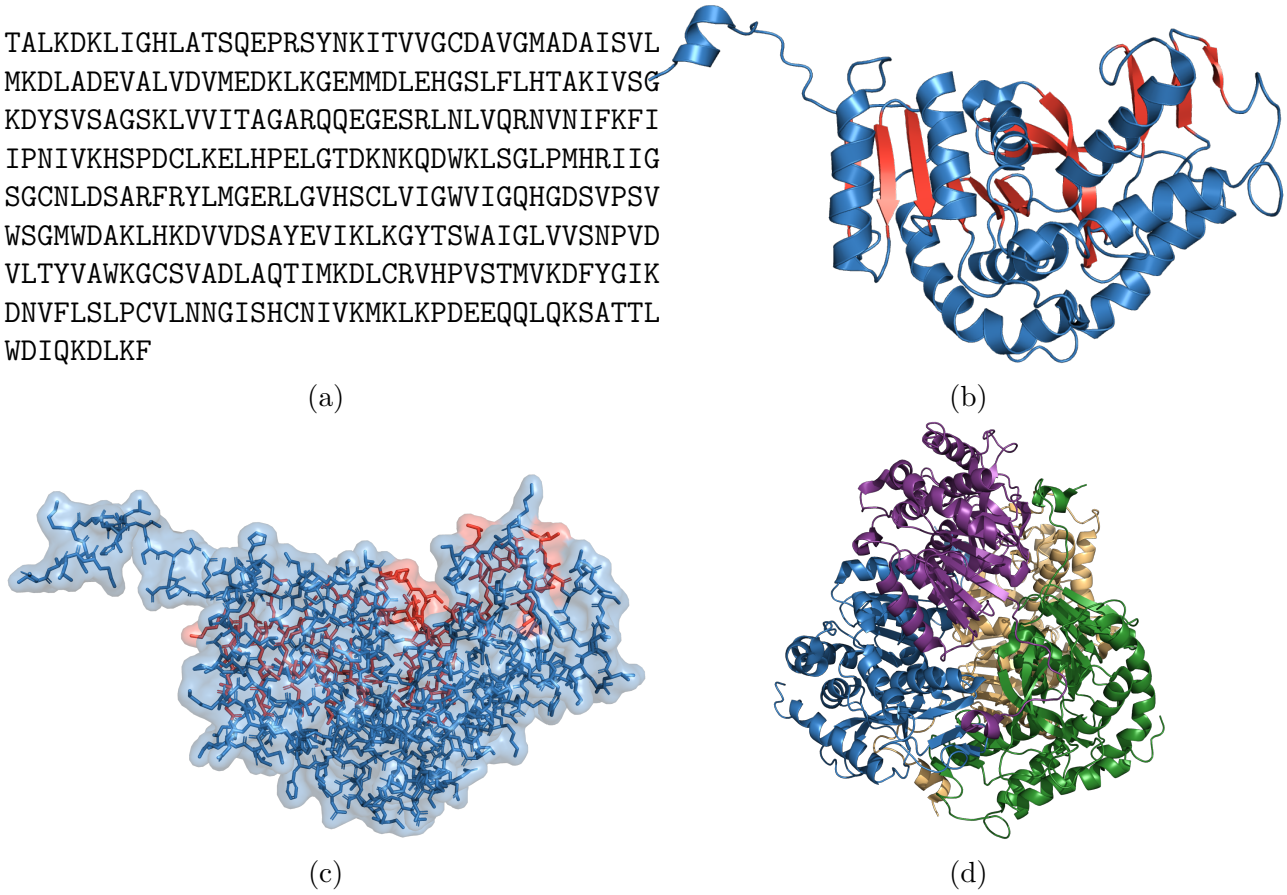


Figure 2.3: Protein structural hierarchy of four levels, for an example protein chain *apo dogfish m4 lactate dehydrogenase* with SCOP ID: d3ldha_ (coming under PDB ID: 3ldh) (a) *Primary structure*: the amino acid sequence, (b) *Secondary structure*: helices (in blue) and strands of sheet (in red), (c) *Tertiary structure*: 3D atomic structure, and (d) *Quaternary structure*: an ensemble of multiple chains to form a functional unit. (Note: molecular visualisations were produced using PyMOL (Schrödinger Inc., 2015))

2.2.1 Evolution of Proteins

The studies of the observed repertoire of proteins across the three domains of life (i.e. Eukarya, Bacteria and Archaea) provide many insights into the mechanisms of macromolecular evolution (Dayhoff, 1978). These proteins have evolved under a complex evolutionary process over billions of years. Research has identified major types of evolution, including divergent evolution and convergent evolution.

Divergent evolution illustrates an event of extant proteins diverging from a common ancestral protein. This results in a homology relationship, forming two different lineages. The pair can represent the same species or two different species if diverged far beyond a speciation event. (In such case they are called orthologs). During this process of evolution, even though the sequences of related proteins undergo drastic change, the core structure tends to be largely conserved while the peripheral regions in their structures refold (Chothia and Lesk, 1986; Lesk, 2010).

Convergent evolution refers to proteins across different lineages, independently evolving to similar end-points, gaining structures that attain similar functions (Pollock et al., 2017).

The focus of this thesis is to study relationships between proteins that have diverged from a common ancestor, thus some general principles that drive divergent evolution of proteins, and their impact on sequence, structure and function are briefly explored below.

Proteins diverge in their sequences due to mutation events on the underlying genetic blueprint, mainly observed as substitutions, insertions and deletions of amino acids. Although structures are more conserved in evolution than sequence, these sequence mutations can also alter the structure and hence affect protein function (Soskine and Tawfik, 2010). Chothia and Lesk (1986) characterise the sequence-structure divergence between related proteins as a function of the fractional sequence identity between their core structures (i.e. $\Delta = 0.4e^{1.87F}$ where F is the fraction of mutated amino acids within the core regions of protein structures). Although this broadly links the divergence of sequence with divergence of structure, it is understood that different loci within proteins can nevertheless have varying rates of change due to varying degrees of structural and functional constraints on them (Lesk, 2001).

As protein sequences diverge, even though there is a pressure to preserve core structure, and consequently function, the changes can also facilitate proteins acquiring new functions, and provide more fitness to adapt to the conditions in which the host organism is evolving. Some examples can be found in the studies of Zahn (2014) and Cheatle Jarvela et al. (2014).

Thus, in protein evolution, the sequence gradually accumulates amino acid substitutions, insertions and deletions. A substitution results in an amino acid being replaced by another amino acid at the same locus. In the seminal work of Dayhoff (1978), this is defined as a *point accepted mutation*. Further, insertions or deletions of amino acids commonly arise due to errors made by molecular machinery during DNA replication or alternative splicing (Miko and LeJeune, 2009; Rosenberg, 2009).

As per the Darwinian *survival of the fittest* (Darwin, 1859), natural selection results from beneficial changes to survive, whereas deleterious changes can cause the organism to be culled from the population. Thus, the basis of divergent evolution is driven by conserving or modifying function (Lesk, 2001).

Commonly, *domains* are considered as independently-evolving structural units of proteins that are conserved in evolution (Marsh and Teichmann, 2010). Proteins acquire new functions via different mechanisms to gain new domains or lose existing domains during evolution. Proteins that share same combinations of domains provide a strong evidence for their divergent evolution (Vogel et al., 2004).

Divergent evolution of a set of protein sequences is often visualised in the form of a rooted phylogenetic tree. Often in such trees, branch lengths at each internal nodes are proportional to the evolutionary time elapsed as sequences diverge from their common ancestors. This time is commonly represented as a function of the observed divergence at the sequence (amino acid) level. To address the subtleties of modelling sequence divergence, several measures have been proposed (Dayhoff et al., 1978; Nei and Kumar, 2000; Kimura, 1983; Ota and Nei, 1994). Previous studies have pointed to an approximately constant rate of amino acid change over any lineage in the phylogenetic tree, as hypothesised in the notion of a *molecular clock* by Zuckerkandl and Pauling (1965). This concept of a molecular clock is based on the idea that the sequences of individual protein families undergo amino acid changes (mainly substitutions) at a constant rate, and different protein families have different such rates (Lesk, 2001). Under this hypothesis, a divergence can be measured in terms of actual time elapsed from their lowest common ancestor (Sarich and Wilson, 1967).

In the inference of phylogenetic trees, extant proteins appear as leaf nodes, whereas internal nodes are hypothesised common ancestors. Accurately inferring such a tree is an extremely challenging computational task (Allison and Wallace, 1994). Several practical methods have been developed to construct phylogenetic trees since early 70s (Fitch, 1971; Felsenstein, 1973; Feng

and Doolittle, 1990; Kishino et al., 1990; Redelings and Suchard, 2005; Brown and Truszkowski, 2012).

2.2.2 Protein Experimental Data Streams and Databases

With major developments in the field of Molecular Biology, the last seven decades have witnessed the development of many Nobel-prize winning technologies to characterise protein sequences, as well as technologies to resolve their 3D structure

Fred Sanger (in 1951) experimentally characterised the chemical signature of Insulin, the first protein to be sequenced, winning him his first of the two Nobel prizes in 1958 (Sanger and Tuppy, 1951). Since then, direct experimental methods such as Edman degradation (Edman et al., 1950) and high-throughput mass spectrometry are major contributors to modern protein sequencing efforts (Saraswathy and Ramalingam, 2011). Further, with the advent of high-throughput genome (DNA) sequencing technologies, first developed in 1970s, it has become feasible to characterise protein sequences by translating *open-reading-frames* within sequenced genomes and then translating the triplets into expected amino acids.

Separately, experimental determination of protein 3D structure started with the development of X-ray crystallographic techniques that led to the resolution of Sperm whale myoglobin and later Human haemoglobin structure. Existing methods to resolve protein 3D structure include X-ray crystallography and Nuclear Magnetic Resonance (NMR) spectroscopy to the recently introduced Cryo-Electron Microscopy (Cryo-EM). However, these methods require a significantly greater amount of experimental effort compared to protein sequencing methods.

Atlas of Protein Sequence and Structure (Dayhoff et al., 1965) was amongst the first efforts to curate protein sequences into databases, with Margaret Dayhoff as the editor. Presently, UniProt database (UniProt Consortium and others, 2017) (`https://www.uniprot.org`) is a publicly available database of protein sequences. At the time of writing this thesis, UniProt contains 195,667,571 amino-acid sequence entries. Currently, UniProt[2] comes under the Protein Information Resource (PIR) (Barker et al., 2000) (`https://proteininformationresource.org`) which is an integrated, public data resource founded in 1984. Protein sequences are exchanged in multiple formats of which, Fast Alignment (FASTA) format introduced by Lipman and Pearson (1985), Protein Information Resource (PIR) format and GenBank format are the most commonly used.

On the other hand, Protein Data Bank (PDB) (Berman et al., 2003) (`http://www.wwpdb.org`) is the primary data source for experimental 3D structures of proteins. At the time of writing, PDB contains structural coordinates of 180,038 proteins and protein-complexes. wwPDB is an umbrella organisation that combines the efforts of the Research Collaboratory for Structural Bioinformatics (RCSB) in USA, PDBe in Europe and PDBj in Japan. Together, they maintain consistency in protein data entries throughout. Atomic coordinate data of proteins are exchanged using either the restrictive Brookhaven Protein Data Bank (PDB) format, or increasingly using more scalable and flexible formats such as the Protein Data Bank Markup Language (PDBML/XML) format, or PDBx/mmCIF (Berman et al., 2003)

2.2.3 Classification of Proteins

Protein classification is a way to organise and catalogue the growing information of proteins (covering sequence, structure and function) to facilitate protein studies and experiments (Koehl,

[2]UniProt database has two sub-repositories: (1) Swiss-Prot for manually-annotated and reviewed protein sequences, and (2) TrEMBL for automatically annotated and unreviewed protein sequences. Currently, there are 563,552 and 195,104,019 entries in them, respectively.

2006; Hadley and Jones, 1999). This classification remains essential to provide reliable benchmarks for validating computational methods to compare proteins.

Proteins are often classified at the level of their *domains*. Richardson (1981) highlighted that, as an independently evolving protein (sub)unit, classifying proteins hierarchically based on their domain(s) remained useful, as many multi-domain proteins are composed of assemblies of domains that resemble domains in other proteins.

Classification of proteins based on the information of their 3D structure and architecture are widely used. Amongst the main domain-level databases in this category are SCOP (Structural classification of proteins (Murzin et al., 1995)) and CATH (Class, Architecture, Topology, Homologous superfamily (Orengo et al., 1997)). Both describe a hierarchical relationship between protein domains. Hadley and Jones (1999) note that they largely provide consistent classification, with SCOP being a good resource for evolutionary information, while CATH provides more on geometric and architectural patterns of folds. This thesis uses SCOP to generate benchmarks for protein comparisons, as discussed in Chapter 5 and 6.

There are four main levels in the SCOP hierarchy: (1) *class*, (2) *fold*, (3) *superfamily* and (4) *family*. Generally, proteins with structural evidence that reflect shared ancestry fall under the same family or superfamily. Proteins that belong to the same family also carry a strong sequence similarity signal as they are closely related. They can be homologs across different species (i.e. orthologs). Stepping up to the next level in the hierarchy, proteins under the same superfamily are distantly related with weak sequence similarity signal, but strong structural and functional conservation. Fold level refers to similar topology observed in the protein domain structures, mainly encompassing the recurrent secondary structural features and their architecture.

A set of proteins that are closely-related are organised into the same *family*. Pfam (Bateman et al., 2000; El-Gebali et al., 2019) is a resource which groups and catalogues by families. Other databases (e.g. InterPro (Apweiler et al., 2001), PANTHER (Mi et al., 2005), PROSITE (Bairoch and Bucher, 1994)) integrate not only the basic sequence data, but also comprehensive function-level details. These databases also inform the functional classification of protein in sync with standard biological schemes such as Gene Ontology (GO)(Ashburner et al., 2000). GO provides functional annotations for proteins in terms of its three hierarchical ontologies: Biological Process, Molecular Function and Cellular Component.

2.3 Protein Comparison via Alignment

Chapter 1 introduced the motivation of protein comparison via alignment, various types of alignments, and the general criteria used to find alignments between proteins. This section elaborates further on the specific technical details of widely-used methods to compute protein sequence alignment.

An *alignment* provides a hypothesis of a potential relationship between proteins by assigning one-to-one correspondences between a subset of their amino acids. In a pairwise alignment, involving two proteins, any alignment relationship $\mathcal{A}$ can be defined as a string over three alignment-states: `match` (`m`), `insert` (`i`), and `delete` (`d`).

A `match` state asserts a relationship between a pair of amino acids. An `insert` state asserts an insertion of an amino acid in (notionally) the second sequence with respect to the first. A `delete` state, often treated symmetrically with `insert` state, asserts a deletion of an amino acid in the first sequence with respect to the second. An illustrative example of an alignment between a pair of amino acid sequences is shown in Figure 2.4a, along with the corresponding alignment states below each alignment column.

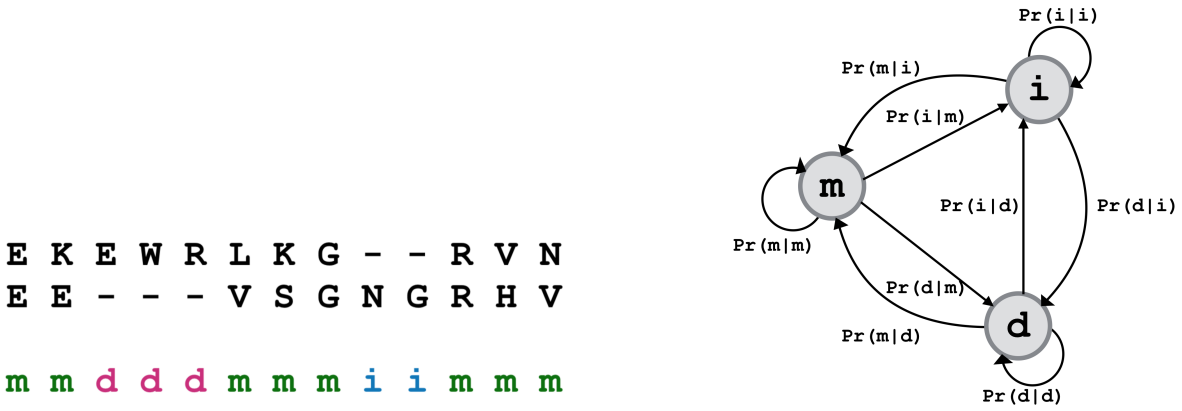


Figure 2.4: (a) Example sequence alignment and corresponding alignment states. (b) Probabilistic finite state machine capable of generating alignment three-state strings.

Consequently, pairwise alignment relationships can be modelled as a string generated from a probabilistic three-state machine illustrated in Figure 2.4b. Such a state machine can generate any three-state string guided by its state-transition probabilities.

The alignment space of a pair of proteins is the set of all possible alignments between them. The size of this space grows factorially as a function of sequence lengths. Given two proteins, $\mathbf{S}$ and $\mathbf{T}$, the minimum possible alignment length is the length of the longest of the two sequences (i.e. $\max(|\mathbf{S}|,|\mathbf{T}|)$). On the other hand, the maximum alignment length is the sum of the two sequence lengths (i.e. $|\mathbf{S}|+|\mathbf{T}|$). Thus, the length $|\mathcal{A}|$ of any possible alignment $\mathcal{A}$ is in the range $[\max(|\mathbf{S}|,|\mathbf{T}|),|\mathbf{S}|+|\mathbf{T}|]$. The total number of possible alignments (i.e. the size of the alignment space) is given by (based on multi-set permutations involving three possible alignment states):

$$\sum_{|\mathcal{A}|=\max(|\mathbf{S}|,|\mathbf{T}|)}^{|\mathbf{S}|+|\mathbf{T}|} \frac{|\mathcal{A}|!}{(|\mathbf{S}|+|\mathbf{T}|-|\mathcal{A}|)!(|\mathcal{A}|-|\mathbf{S}|)!(|\mathcal{A}|-|\mathbf{T}|)!}$$

Any alignment can be visualised as a source-to-sink path in a matrix of order $(|\mathbf{S}|+1)\times(\mathbf{T}+1)$, where the source denotes the cell $(0,0)$ and sink $(\mathbf{S},\mathbf{T})$.

In general, the search for meaningful alignment relationships between proteins in this large space of alignments is posed as an optimisation problem. Sequence alignment methods generate an alignment solely based on amino acid sequence information. Common criteria used to quantify a sequence alignment involve a substitution scoring matrix and gap penalty function. Many substitution matrices and gap penalty functions have been developed. As the main focus of this thesis, the commonly-employed methodological aspects of computing protein sequence alignment are discussed in full detail in §2.4.

It is well studied that amino acid sequences diverge more drastically in evolution compared to their 3D structures (Chothia et al., 2003). However, in the absence of known structure, which remains time-consuming and hard to resolve experimentally, sequence provides the only available recourse to comparing proteins and glean their relationships. However, studies have shown that the detection of these relationships becomes increasingly difficult as proteins diverge (Vogt et al., 1995; Rost, 1999; Do et al., 2005; Habermann, 2016). Detecting reliable relationships between *twilight zone* sequences – these are said to be sequences whose percentage identity of amino acids is below 35% – remains a challenging problem (Doolittle, 1986; Habermann, 2016).

The extreme limits to a viable inference of an alignment relationship using sequence information alone has been hypothesised to $\sim$ 10% amino acid sequence identity, which is almost a value expected for two randomly generated protein sequences (Rost, 1999, 1997).

Thus, detection of relationship between highly-diverged sequences, termed *remote orthology detection*, is an important and challenging task. For example, human orthologs of *Saccharomyces cerevisiae* are examined to understand many mitochondrial diseases (Barrientos, 2003; Szklarczyk et al., 2012). Note that *Homo sapiens* and *Saccharomyces cerevisiae* diverged approximately one billion years ago and yet, they carry a considerable number of functionally similar proteins (Douzery et al., 2004; Kachroo et al., 2015). Thus, capturing such remote relationships is salient.

With the above motivation, this thesis will explore the widely used methods and criteria to address the protein sequence alignment problem.

2.4 Protein Sequence Alignment

Protein sequence alignment is a classical computational problem, with a large corpus of methods and programs available to address it. The widely-used criteria and methods used to compute sequence alignments are discussed below.

2.4.1 General Sequence Alignment Scoring Criteria

Commonly, the similarity of matched pairs of amino acids in sequence alignments is quantified using scores derived from a (user-specified) substitution scoring matrix. Separately, the unmatched regions of the alignment appearing as inserted runs of amino acids in one sequence or deleted runs of amino acids in the other, termed as *gaps*, are penalised using a run-length dependent gap-penalty function, whose parameters (typically, gap-open and gap-extension penalties) are also user-specified, if not the default values that come specified for any chosen substitution scoring matrix. These two aspects of quantifying an alignment are elaborated below.

Scoring matches using log-odds scores from a substitution scoring matrix

Each match in an alignment three-state string is describing corresponds to one of the $20 \times 20 = 400$ possible ways two amino acids can be matched. A typical substitution matrix $\mathbf{L}$ is therefore a 20×20 matrix with each cell (i, j) storing the score of matching a pair of amino acids indexed by i (aa_i) and j (aa_j). Conventionally, a substitution between any pair of amino acids is treated to be directionless (i.e. $\mathbf{L}_{ij} = \mathbf{L}(aa_i, aa_j) \equiv \mathbf{L}(aa_j, aa_i) = \mathbf{L}_{ji}$) (George et al., 1990).

Specifically, each score between any pair of amino acids in the traditional substitution matrices is defined as a log-odds ratio. It is the logarithm of the ratio between the probability of the pair being related and the probability that they are unrelated, under some probabilistic model. The odds ratio gives the ratio of their joint occurrence probability, $\Pr(aa_i,aa_j)$, to their independent occurrence probability, $\Pr(aa_i)\cdot\Pr(aa_j)$. Therefore, the log-odds ratio is of the form:

$$\mathbf{L}(aa_i, aa_j) = \log\left(\frac{\Pr(\langle aa_i, aa_j\rangle)}{\Pr(aa_i)\cdot\Pr(aa_j)}\right)$$

Interestingly, the log-odds scores can be rationalised in information-theoretic terms. In general, using the mathematical theory of communication of Shannon (1948), the information content of an event E with probability Pr(E) (under some probabilistic model) is given by

$\mathrm{I(E)} = -\log_2(\Pr(\mathrm{E}))$ bits. This value denotes the shortest length of a decodable codeword to encode (i.e., *explain*) the event E under that model. Applying this insight, the log-odds score can be expanded as:

$$\mathbf{L}(aa_i, aa_j) = \log\left(\frac{\Pr(\langle aa_i, aa_j\rangle)}{\Pr(aa_i)\cdot\Pr(aa_j)}\right) = \underbrace{I(aa_i) + I(aa_j)}_{\text{independent explanation}} - \underbrace{I(\langle aa_i, aa_j\rangle)}_{\text{joint explanation}}$$

The sum of first two terms on the right hand side, $\mathrm{I}(aa_i) + \mathrm{I}(aa_j)$, represents the length of encoding each amino acid independently. The last term $\mathrm{I}(\langle aa_i, aa_j\rangle)$ represents the length of encoding the information of the amino acids jointly. The difference denotes the compression gained/lost due to their joint explanation compared to an independent explanation.

Thus, the positive scores in any log-odds matrix reflect favourable substitutions, whereas negative scores indicate unfavourable substitutions (Altschul et al., 1990). Many substitution scores are often linearly scaled prior to their use, mainly to represent the (scaled) log-odds scores as integers within a range. A positive multiplicative factor c used to scale the log-odds scores, translates to the measurement of compression in $1/c$ bit units. For instance, +12 score for a pair in a substitution scoring matrix with scores scaled to one-fourth-bit units conveys that the related model for the corresponding amino acid pair makes the data 2^3 times more likely than the unrelated model. Note, scaling of log-odds score does not affect the optimisation of the alignment score. Applying a positive multiplier or adding a constant per character to an alignment score/cost value does not change the rank-order of a possible alignment(Allison, 1993).

Note: Summaries of nine widely-used substitution scoring matrices are provided in Chapter 6.

Penalising Insertions and Deletions using Gap Penalty Functions

Insertions and deletions (gaps) are handled using a length-dependent penalty function (denoted by $\Gamma(l)$). Widely reported gap penalty functions are *linear*, piecewise-linear (or *affine*), and *concave* gap functions.

Using a *linear* gap function, each insertion and deletion is penalised by a constant amount g. *Affine* gap function penalises a gap over two parts: a (stringent) penalty for opening a gap, controlled by the parameter g_{open}, and a (less stringent) penalty for extending an existing gap, controlled by the parameter g_{extend}, which grows linearly with the length of the extension. *concave* gap penalty function has the same consideration for opening a gap, but the penalty for the entire gap grows logarithmically with the gap length. Table 2.2 characterises all these three gap functions.

Both affine and concave gap functions are considered more biologically meaningful, as they attempt to capture the insight that insertions and deletions are more likely to appear *en bloc* in protein sequences, given that nucleotide-level insertions and deletions also appear in blocks (Cartwright, 2006). Thus, these two gap penalty functions ensure that the observed

Table 2.2: Commonly-used gap penalty functions

Type	**Gap penalty function**
Linear	$\Gamma(l) = g \times l$
Affine	$\Gamma(l) = g_{open} + g_{extend}\times(l-1)$
concave	$\Gamma(l) = g_{open} + g_{extend}\times\log(l)$

gaps in any alignment are realistic and avoid favouring short stretches of gaps, interspersed between the matches in any alignment.

From a point of view of alignment three-state machines, the penalty for any gap of length l under the *affine* gap function can be rationalised in terms of the logarithm of the probability of a random variable over a geometric distribution. Given a probability of opening a gap as Pr(gap-open) and the probability of extending a gap as Pr(gap-extension), the length of the gap can be modeled as a random variable over a geometric distribution with the parameter Pr(gap-extension). In general, the geometric distribution models the number of Bernoulli trials needed to observe a certain outcome. Here the outcome is, ending the current gap. Therefore, under this model, the likelihood of any gap with length l is given by:

$$\Pr(\text{gap-open}) \times (\Pr(\text{gap-extension}))^{l-l}.$$

The logarithm of this likelihood yields a form similar to the affine gap penalty function:

$$\log(\Pr(\text{gap-open})) + (l-1) \times \log(\Pr(\text{gap-extension}))$$

Note, that the expected length of any geometrically distributed gap length (random variable) is given by:

$$\frac{1}{[1 - \Pr(\text{gap-extension})]}$$

The *affine* gap model is more widely used than the *concave* scheme. This is because, the alignments can be computed more efficiently (i.e. computational complexity wise) under the affine gap model compared to the latter (Gotoh, 1982; Rivas and Eddy, 2015).

Finally, it is common practice for users to choose a substitution scoring matrix and accompanying gap parameters when aligning protein sequences.

2.4.2 Searching for the Best Alignment

Dynamic programming (DP) (Bellman et al., 1954) is commonly used to address the problem of sequence alignment. DP is applicable when an optimisation problem exhibits two key characteristics: (1) an optimal substructure and (2) overlapping set of subproblems. An optimal substructure gives the ability to partition a larger problem into smaller subproblems such that the optimal solution of the larger problem can be constructed using the optimal solutions for the subproblems. The subproblems are said to be overlapping when larger problems share a number of smaller subproblem instances, thus permitting each subproblem instance to be explicitly evaluated exactly once, and then *memoised* (remembered) and recalled as and when the subproblem solution is needed again.

The sequence alignment problem, using a substitution scoring matrix and any of the three gap penalty functions discussed in the previous section, exhibits both these properties, permitting the application of a DP algorithm (Needleman and Wunsch, 1970; Gotoh, 1982; Lipman and Pearson, 1985). Specifically, considering the alignment of two proteins $\mathbf{S}$ and $\mathbf{T}$:

Optimal substructure: The alignment of the prefixes $\mathbf{S}_{1..i}$:$\mathbf{T}_{1..j}$ of the two protein sequences depends on the optimality of its subproblems: (1) the alignment of $\mathbf{S}_{1..i-1}$:$\mathbf{T}_{1..j-1}$ (followed by a match of the amino acids $\mathbf{S}_i$:$\mathbf{T}_j$), (2) the alignment of $\mathbf{S}_{1..i}$:$\mathbf{T}_{1..j-1}$ (followed by a deletion of the amino acid $\mathbf{T}_j$) and (3) the alignment of $\mathbf{S}_{1..i-1}$:$\mathbf{T}_{1..j}$ (followed by an insertion of the amino acid $\mathbf{S}_i$).

Overlapping subproblems: In the above subproblem structure, many subproblems overlap. For instance, the subproblems $\mathbf{S}_{1..i}$:$\mathbf{T}_{1..j}$ and $\mathbf{S}_{1..i+1}$:$\mathbf{T}_{1..j+1}$ share the same optimal solutions of subproblems $\mathbf{S}_{1..k}$:$\mathbf{T}_{1..l}$ $\forall 1 \leq k \leq i, 1 \leq l \leq j$. These overlapping subproblems are computed exactly once and reused whenever needed later on.

The earliest application of DP for string comparison was by Levenshtein (1966) who introduced the notion of *edit distance* (now sometimes called Levenshtein distance). For biological sequence comparison, Needleman and Wunsch (1970) and Smith et al. (1981) applied a Dynamic Programming Algorithm (DPA) to address the *global* sequence alignment and *local* sequence alignment, respectively.

In the mechanics of aligning sequences $\mathbf{S}$ and $\mathbf{T}$ using a DPA, a *memoisation* (history) matrix, denoted by Hist, is maintained, of size $(|\mathbf{S}|+1 \times |\mathbf{T}|+1)$. Any cell $\text{Hist}(i,j)$ records the optimal alignment score between the prefixes $\mathbf{S}_{1..i}$ and $\mathbf{T}_{1..j}$.

As we saw above, there are three ways to derive the optimal alignment at cell (i,j): (1) using the optimal value in the cell $(i-1,j-1)$ to compute the optimal alignment ending in a match between the symbols $\mathbf{S}_i$ and $\mathbf{T}_j$, (2) using the optimal value in the cell $(i,j-1)$ to compute the optimal alignment ending in a deletion of symbol $\mathbf{T}_j$, or (3) using the optimal value in the cell $(i-1,j)$ to compute the optimal alignment ending in an insertion of symbol $\mathbf{S}_i$.

Note, an alignment problem can be formulated as maximising the alignment score or minimising the alignment cost/distance. This section explores various formations of the DPA as a *maximisation problem*, given a substitution matrix $\mathbf{L}$ and varying gap penalty functions.

Using a linear gap penalty function of the form $\Gamma(l) = \mathbf{g} \times l$, the DPA can be formalised using following recurrence relationships:

$$\text{Hist}(i,j) = \max \begin{cases} \text{Hist}(i-1,j-1) + \mathbf{L}(\mathbf{S}_i, \mathbf{T}_j) \\ \text{Hist}(i-1,j) + g \\ \text{Hist}(i,j-1) + g \end{cases}$$

where $\mathbf{L}(\mathbf{S}_i, \mathbf{T}_j)$ is the substitution score for matching the two amino acids $\mathbf{S}_i$ and $\mathbf{T}_j$, and $\mathbf{g}< 0$ is a per-symbol gap penalty parameter under the linear gap model.

In terms of algorithmic complexity, this DP algorithm requires $O(|\mathbf{S}||\mathbf{T}|)$ time and space to compute. Once the matrix filling is complete, an optimal alignment can be generated by tracing back the optimal derivations made at each cell, starting from $(|\mathbf{S}|, |\mathbf{T}|)$ to $(0,0)$, which takes an additional worst-case time of $O(|\mathbf{S}|+|\mathbf{T}|)$.

A general gap penalty function $\Gamma(\cdot)$ is said to be *sub-additive* when $\Gamma(l_1 + l_2 + \ldots + l_n) < \Gamma(l_1) + \Gamma(l_2) + \ldots + \Gamma(l_n)$ $\forall l_i > 0$. The concave gap penalty function and the affine gap penalty function are both sub-additive. Thus, using these gap penalty functions and a substitution matrix $\mathbf{L}$, a general DPA recurrence can be defined by exploring gaps of varying lengths for each subproblem, as follows (Needleman and Wunsch, 1970; Gusfield, 1997; Miller and Myers, 1988):

$$\text{Hist}(i,j) = \max \begin{cases} \text{Hist}(i-1,j-1) + \mathbf{L}(\mathbf{S}_i, \mathbf{T}_j) \\ \max_{k=1}^{i} \text{Hist}(i-k,j) + \Gamma(k) \\ \max_{k=1}^{j} \text{Hist}(i,j-k) + \Gamma(k) \end{cases}$$

A straightforward implementation of these recurrences results in the worst-case run-time complexity of $O(|\mathbf{S}|^2|\mathbf{T}| + |\mathbf{S}||\mathbf{T}|^2)$. However, Miller and Myers (1988) have proposed a faster $O(|\mathbf{S}|^2 \log(|\mathbf{T}|) + \log(|\mathbf{S}|)|\mathbf{T}|^2)$ time algorithm.

Specifically for affine gap function, Gotoh (1982) presented a time-efficient DP recurrence. It utilises the three-state nature of an alignment over match (m), insert (i) and delete (d) states. In this DPA, the alignment of prefixes (subproblems) ending in each of these states are represented using three different history matrices: $\text{Hist}_{\texttt{m}}$, $\text{Hist}_{\texttt{i}}$ and $\text{Hist}_{\texttt{d}}$. That is, each cell (i, j) of matrix $\text{Hist}_{\texttt{m}}$ corresponds to an alignment of prefix $\mathbf{S}_{1..j}$ and $\mathbf{T}_{1..i}$ ending in a match state. Similarly, $\text{Hist}_{\texttt{i}}(i, j)$ and $\text{Hist}_{\texttt{d}}(i, j)$ relates to those prefixes ending in an insertion and a deletion, respectively. Thus, the affine gap penalty function can be employed using the following DP recurrence relation for each of the three matrices.

$$\text{Hist}_{\texttt{m}}(i,j) = \max \begin{cases} \text{Hist}_{\texttt{m}}(i-1, j-1) + \mathbf{L}(\mathbf{S}_i, \mathbf{T}_j) \\ \text{Hist}_{\texttt{i}}(i-1, j-1) + \mathbf{L}(\mathbf{S}_i, \mathbf{T}_j) \\ \text{Hist}_{\texttt{d}}(i-1, j-1) + \mathbf{L}(\mathbf{S}_i, \mathbf{T}_j) \end{cases}$$

$$\text{Hist}_{\texttt{i}}(i,j) = \max \begin{cases} \text{Hist}_{\texttt{m}}(i-1, j) + g_{open} \\ \text{Hist}_{\texttt{i}}(i-1, j) + g_{extend} \\ \text{Hist}_{\texttt{d}}(i-1, j) + g_{open} \end{cases}$$

$$\text{Hist}_{\texttt{d}}(i,j) = \max \begin{cases} \text{Hist}_{\texttt{m}}(i, j-1) + g_{open} \\ \text{Hist}_{\texttt{i}}(i, j-1) + g_{open} \\ \text{Hist}_{\texttt{d}}(i, j-1) + g_{extend} \end{cases}$$

As with the linear model, DPA with affine gap function requires $O(|\mathbf{S}||\mathbf{T}|)$ space and time, although with a larger constant factor.

Smith et al. (1981) modified the global sequence alignment algorithm of Needleman and Wunsch (1970) to address a variant of this problem, termed the local alignment problem, of identifying subsequences of proteins that are related. Consequently, the DPA requires a minor alteration in the recurrence relations that enables identifying such local regions. This variation enforces the algorithm to always select a non-negative alignment score at each cell, by introducing an additional case of a 0 alignment score. While global alignment finds an overall optimal alignment, possibly with longer stretches of low similarity regions aligned, local alignment aims for locally conserved regions by optimising in the local vicinity (Altschul et al., 1990; Needleman and Wunsch, 1970; Smith et al., 1981). The trace-back to obtain the local alignment starts from the cell with the maximum score until a cell with 0 score is reached.

Test for Alignment Significance

Alignment scores require a test statistic for alignment significance to differentiate between related and unrelated proteins (Durbin et al., 1998). The key question is: *how likely it is to see a certain optimal alignment score by chance?*. This calls for a classical statistical significance test. A null model is required for this purpose (i.e. the model in which the proteins are deemed unrelated with each other). Thus, the criterion targets to evaluate the likelihood of an alignment score under a null model. Commonly, *P-value* statistic is used for this purpose, attempting to estimate the probability of receiving an alignment score at least as extreme as under a null model distribution of scores.

Karlin and Altschul (1990) published some important statistics to allow for an empirical model based statistical significance test for local alignment. They found that the distribution of optimal local alignment scores for ungapped alignments between unrelated sequences approximately follows a Gumbel distribution.[3] It builds on the following approximation: Given

[3]The Gumbel distribution is a type of skewed continuous probability distribution coming under the family of Extreme value distribution, with two parameters defining its probability density function.

two random sequences with lengths m and n, the number of local alignments with a score $x \geq S$ is said to be Poisson distributed with a mean statistic of $Kmne^{-\lambda S}$ with two parameters $K, \lambda > 0$. This is known as the *e-value* which gives the expected number of distinct optimal local alignments with a score of at least S (Altschul et al., 2001). The smaller the *e-value* value is, the more statistically significant the local alignment score S is. Under the Gumbel distribution, the *p-value* provides the probability of observing any score x greater than the score S is $\Pr(x \geq S) \approx 1 - e^{Kmne^{-\lambda S}}$. This has become a basis for reporting statistical significance of local alignments (e.g. in BLAST (Altschul et al., 1990) and FASTA (Pearson and Lipman, 1988)). Later, similar observation was noted for gapped local alignments by Levitt and Gerstein (1998), as well as for global alignments by Bastien and Maréchal (2008).

2.4.3 Probabilistic Hidden Markov Model for Sequence Alignment

A Pair Hidden Markov Model (HMM) provides a rigorous approach to address the pairwise alignment method. This approach uses the three-state machine to formulate the task as a hidden Markov process in probabilistic terms. The concepts underlying HMMs have been comprehensively discussed by Durbin et al. (1998), and been applied in many HMM based alignment programs (e.g. HMMER (Eddy, 1998) and ProbCons (Do et al., 2005)). They also proposed ways to generate alternate alignments rather than just optimal ones. A comprehensive discussion of pair HMMs and their correspondence with conventional alignment scoring has been presented in the recent work of Frith (2020). The core ideas that Pair HMM utilises arose earlier in the MML literature on DNA sequence alignment problem due to the work of Allison and Wallace (1994); Yee and Allison (1993).

Pair HMM defines a state space $\Omega = \{\texttt{m}, \texttt{i}, \texttt{d}\}$, an initial state probability vector, a state transition probability matrix, the observed symbol space over the 20 amino acid types, and their emission probability matrix. The state `m` emits a pair of amino acids, whereas states `i` and `d` emit amino acid symbols for the respective gap regions. The next state to transmit and the current emission both depend only on the current state (making the process *memoryless*). Accordingly, any state sequence generated through this HMM model is an alignment path which can describe the observed pair of sequences. Given a fully parameterised pair HMM model and two observed sequences, $\mathbf{S}$ and $\mathbf{T}$, an inference problem involves two independent tasks: (1) *scoring* and (2) *decoding*.

Scoring: refers to the evaluation of a state sequence (an alignment path) under the given HMM model. This includes:

- **Single-path scoring**: the joint probability of an alignment π and the sequences:

$$\Pr(\pi, \mathbf{S}, \mathbf{T})$$

- **All-path scoring**: the full joint probability of the sequences summing over all possible alignment paths (i.e. marginal probability):

$$\Pr(\mathbf{S}, \mathbf{T}) = \sum_{\pi} \Pr(\pi, \mathbf{S}, \mathbf{T})$$

  This can be computed by applying the *forward algorithm*: a dynamic programming procedure to compute $\Pr(\mathbf{S}_{1..i}, \mathbf{T}_{1..j}, \pi_t = l)$: the joint probability of the corresponding pair sequence prefixes and any alignment path between them which ends in state l, leading to $\Pr(\mathbf{S}, \mathbf{T}, \pi_{|\pi|} = l)$ for any state $l \in \Omega$. Finally, $\Pr(\mathbf{S}, \mathbf{T}) = \sum_{l \in \Omega} \Pr(\mathbf{S}, \mathbf{T}, \pi_{|\pi|} = l)$.

Decoding finds the best path that describes the observed pair of sequences. This includes:

- **Single-path decoding:** defines the the most probable alignment path that optimises the single-path joint probability:

$$\underset{\pi}{\operatorname{argmax}} \Pr(\pi, \mathbf{S}, \mathbf{T})$$

  This is obtained using the *Viterbi algorithm*, a dynamic programming algorithm that takes $\underset{l \in \Omega}{\operatorname{argmax}}\{\Pr(\mathbf{S}_{1..i}, \mathbf{T}_{1..j}, \pi_t = l)\}$ into account.

- **All-path decoding/posterior decoding:** results in the path π which has the most likely state for each position t in the state string, by computing:

$$\underset{k \in \pi}{\operatorname{argmax}} \Pr(\pi_t = k | \mathbf{S}, \mathbf{T})$$

  This posterior probability can be computed using *forward-backward* procedure, where

$$\Pr(\pi_t = k, \mathbf{S}, \mathbf{T}) = \underbrace{\Pr(\mathbf{S}_{1..i}, \mathbf{T}_{1..j}, \pi_t = k)}_{forward} \underbrace{\Pr(\mathbf{S}_{i+1..|\mathbf{S}|}, \mathbf{T}_{j+1..|\mathbf{T}|} | \pi_t = k, \mathbf{S}_{1..i}, \mathbf{T}_{1..j})}_{backward}$$

  for $\Pr(\pi_t = k | \mathbf{S}, \mathbf{T}) = \frac{\Pr(\pi_t = k, \mathbf{S}, \mathbf{T})}{\Pr(\mathbf{S}, \mathbf{T})}$.

  The all-path decoding strategy is applied to find $\Pr(\pi_t = \texttt{match} | \mathbf{S}, \mathbf{T})$ the probability of matching a certain pair of amino acids: $\mathbf{S}_i, \mathbf{T}_j$, given the entire two sequences, using the above *forward-backward* procedure and the full joint probability of sequences. As a result, it allows searching for the alignment path which maximises this posterior probability for each residue pair (consequently maximising the probability of the state sequence). Separately, $\Pr(\pi_t = \texttt{insert} | \mathbf{S}, \mathbf{T})$ can be considered as well. However, this does not guarantee a well-formed feasible alignment (Eddy, 1998). Instead, an alternative method has been presented for maximising the expected accuracy of an alignment. Given an alignment π, the expected accuracy is defined as:

$$A(\pi) = \sum_{(i,j) \in \pi} \Pr(\mathbf{S}_i \text{ matches } \mathbf{T}_j)$$

  This is simply the sum over all aligned pairs represented by the alignment string. The maximisation is performed by following the conventional dynamic programming algorithm with following DP recurrence relations.

$$M(i,j) = \max \begin{cases} M(i-1, j-1) + \Pr(x_i \text{ matches } y_j) \\ M(i-1, j) \\ M(i, j-1) \end{cases}$$

  Nonetheless, it disregards the possible consideration of any unspecified state (i.e. which could be either a `match`, `insert`, or `delete`).

Parameters of a pair HMM model is usually inferred either through an Expectation-Maximisation approach such as Baum-Welch and Viterbi training, or upon a known dataset of pairwise alignments (Allison et al., 1992b; Eddy, 1998; Truszkowski and Brown, 2011). Further HMM methods are notably discussed by Holmes (1998), highlighting the importance and incorporation of evolutionary time-dependent substitution models and gap models.

In addition to the generative pair HMM probabilistic model, some have also looked at applying a discriminative model such as pair conditional random field (e.g. CONTRAlign (Do et al., 2006)). This is a notable method for pairwise protein alignment by Do et al. (2006) as they implemented an unsupervised parameter estimation by optimising both substitution matrix and gap penalties, effectively over a very few example alignments. They additionally aims for a trade-off between model fit and complexity, by controlling the complexity through a set of regularisation parameters.

Finally, at a high-level, the above probabilistic methods share broad similarities with the general approach explored in this thesis. In other words, the pair HMM methods provide the potential to infer probabilistic parameters of alignments, explore sub-optimal (competing) alignments and compute marginal probability models. However, the similarities end there. The minimum message length framework and the rigour it provides for inductive inference make this attempt markedly different from the above, and relies on ideas that link statistical inference with lossless data compression. Furthermore, the MML parameter estimation and the way complexity-versus-fit trade-off is addressed distinguishes this attempt considerably. In fact this thesis is a continuation of the line of research originally proposed for DNA sequence alignment by Allison et al. (1992b); Yee and Allison (1993); Allison and Wallace (1994); Powell et al. (2004), and extended here to protein sequences along with the unsupervised inference of a complete set of statistical models supporting this formulation (discussed in Chapter 4).

2.4.4 Database Search

With the advent and rapid progression of protein sequence databases, there is a need to swiftly search over all sequences in a target database, to find similar sequences to a query sequence. Running a pairwise DPA between the query and every sequence in the database comes with a heavy computational effort. Therefore, heuristic approaches aim to approximate an optimal alignment, and reduce the computational effort by orders of magnitude (Altschul et al., 1990). Methods to search large sequence databases make a trade-off between speed and sensitivity (Altschul et al., 1997). Often these methods rely on identifying significant seeds to construct local, ungapped alignments. Such seed represents a diagonal segment in the dot-plot between the sequences in comparison. Below summarises two popular heuristic search algorithms, mainly for local alignment.

Basic Local Alignment Search Tool (BLAST) (Altschul et al., 1990)

BLAST defines a segment pair for seeding. A segment pair is a same length substring alignment of contiguous residue stretches amongst two sequences. A high scoring segment pair (HSP) refers to a one with a significantly higher score, indicating locally conserved regions. Two proteins may have several HSPs. A maximal segment pair (MSP) refers to the one that cannot be improved in its score by further extension or shortening. BLAST heuristically searches for all locally maximal segment pairs above a predefined threshold score. The process first breaks down a query protein into k-mers of length 3 (3-mer). Next, using a scoring matrix, each 3-mer is scored against all of the possible 3-mer combinations comprised of the three amino acid residues present in it (including the identical match 3-mer). Only the matched 3-mers

having at least some threshold score is kept in record. Then, each target sequence is scanned against each recorded 3-mer for an exact match. Originally, BLAST takes such match as a seed for a possible ungapped alignment and extends it in both directions along the sequence until the segment score stops increasing further and starts dropping under some predefined value compared to the maximum score so far. An extended seed is accepted and reported as an HSP only if it is statistically significant. The *e-value* as described in §2.4.2 is useful for this, with the length parameters being the query sequence length and the sum length of all sequences in the database. In case of multiple HSPs, each is reported as an individual, ungapped local alignment. Later version ((Altschul et al., 1997; Tatusova and Madden, 1999)) aggregates all accepted HSPs to form a single, gapped alignment. Further modification joins seeds within a short distance on the same diagonal to be a single seed, and extend it further for getting an HSP. Next banded DP is applied to obtain a single, gapped alignment containing all accepted HSPs (Altschul et al., 1997; Mount, 2007). Later, several other variants of BLAST including the Position-Specific Iterative BLAST (PSI-BLAST) were also introduced.

FASTA (Pearson and Lipman, 1988)

FASTA was introduced along with the legacy protein sequence file format FASTA (FAST-All). It is a four step search algorithm for scoring pairwise sequence similarity with a parameter called *ktup*. First it finds identically matched regions of length *ktup* between two protein sequences (Note: *ktup* is 1 or 2 for proteins) using a look-up table. The best diagonal regions of matches are selected based on the number of matches and their distances. Next, the selected matches are scored using a scoring matrix. Matches with scores greater than a predefined threshold score are then joined optimally. Finally, dynamic programming based global or local alignment is performed over a banded region centered around the highest scoring initial region.

2.4.5 Multiple Alignment

Multiple protein sequence alignment targets an alignment of more than two proteins, generalising the pairwise alignment problem. This is useful in constructing representative profiles for families of related proteins, and phylogenetic trees. A multiple alignment can be scored similarly as in pairwise alignment, by summing up scores for all possible pairs. However, dynamic programming for score optimisation becomes computationally expensive and intractable as the number of sequences grow more than two. Therefore many current methods use heuristics for the purpose (Venclovas, 2011). One common approach is progressive alignment (Hogeweg and Hesper, 1984; Feng and Doolittle, 1990). The general idea is to first compute alignment scores for all possible sequence pairs, forming a distance matrix. Based on that, a guide tree is built using a hierarchical clustering method. The tree then directs progressive alignment, starting from most similar pairs to least similar pairs. Idea is to take each node of the tree in the order of their insertion to the tree, and align their child nodes (Durbin et al., 1998). Well known progressive sequence alignment algorithms include ClustalW (Larkin, 2007), MAFFT (Katoh et al., 2002) and T-Coffee (Notredame et al., 2000). MUSCLE was suggested by Edgar (2004) for an iterative refinement of alignments, to address the propagation of error during each step of progressive alignment. Other methods include profile HMM based models (e.g. HMMER (Eddy, 1998)).

2.5 Limitations in Sequence Alignment

Below discusses the key limitations and shortcomings we see in the present state of the art of protein sequence alignment.

Ad hoc parameter choices

In common practice, alignment programs are used with *ad hoc* parameter choices, and yield radically different results under different parameter settings. As discussed in §2.4.1, sequence alignments depend on the selection of (1) a substitution scoring matrix, and (2) a gap penalty function. However, the choice of the scoring matrix is often made with little or no attention given to the *divergence* of sequence-pairs that are being aligned. In principle, the degree of evolutionary divergence should be amongst the parameters inferred *during* the alignment process. Instead, most users resort to using a 'general-purpose' substitution matrix, for example BLOSUM-62 (Henikoff and Henikoff, 1992) or PAM-250 (Dayhoff et al., 1978). To compound the problem, the parameters of the chosen gap penalty function is another source of arbitrariness. In practice, users are left to fiddle with these penalty parameters and make them work with their chosen substitution matrix. Previous studies have highlighted this as a "trial and error based exercise" (Do et al., 2005; Vingron and Waterman, 1994). It is not entirely uncommon for alignment programs to be driven on default gap parameters that come with a chosen substitution matrix. Such decisions are at best empirical, if not anecdotal (based on the reported values in the literature) or arbitrary.

Indeed, finding the optimal settings for alignment runs is challenging. Previous studies have sought to systematically explore the space of alignment parameters when searching for an optimal alignment (Vingron and Waterman, 1994; Barton and Sternberg, 1987; Fitch and Smith, 1983; Blake and Cohen, 2001). However, along with the other drawbacks (see below) in the objective function that is commonly optimised, an exhaustive search is inefficient.

Handling of 'complexity-versus-fit' trade-off

The general class of inference problems in which one infers propositions/hypotheses on observed data necessarily have to address the trade-off between the complexity of the proposition and its fit to the data. It is observed that the alignment problem is an instance in this class of problems. Hence it is unavoidable for any method to infer alignment relationships on observed sequence data to deal with how complex an alignment is versus how well it describes the data. In the common formulations of the sequence alignment objective function that relies on substitution *scores* and gap *costs*, this score-versus-cost scheme provides only an unprincipled and approximate way of addressing the aforementioned trade-off. A more systematic way of addressing this trade-off is a desirable property for any alignment framework to possess, and hence is one of the shortcomings.

Disconnect between models addressing substitutions and models addressing insertions and deletions

Another major lacuna is the disconnect between models for amino acid substitutions and those handling insertions and deletions. Gonnet et al. (1992) emphasises the importance of using matrices suitable for the optimal evolutionary time between the proteins being compared, for each individual run. Markov models (matrices) of amino acid substitutions (E.g. PAM (Dayhoff et al., 1978)) suggest an elegant way to model protein sequence evolution through substitutions, and implicitly define a notion of divergence; however the choices of gap penalty parameters to

go with these matrices are usually hand-tuned without solid mathematical support (Blake and Cohen, 2001; Vogt et al., 1995; Gonnet et al., 1992; Chang and Benner, 2004; Benner et al., 1993). Holmes (1998) discusses the value of incorporating substitution models and gap models parameterised on evolutionary time in the context of HMMs. A reliable set of statistical models that are able to directly connect the models of amino acid substitutions with models that address insertions and deletions, and adapt in concert to address alignments across the spectrum of sequence relationships would be a useful milestone in the sequence alignment state of the art.

Lack of a framework to systematically explore competing alignments

Alignment programs often report only a single optimal alignment, overlooking other closely-competing alignments. Previous studies have acknowledged the importance of exploring the sub-optimal alignment space or taking all possible alignments into account, suggesting a few approaches to overcome this scenario (Rosenberg, 2009; Durbin et al., 1998; Rost, 1999; Redelings and Suchard, 2005; Do et al., 2005; Levy Karin et al., 2018). However, the optimal alignment under current objective functions and ad hoc parameter choices makes methodical explorations of closely-competing alignments and sequence relationships very difficult, if not impossible. (Note: An optimal alignment under some objective function may not always be the one that captures the true evolutionary relationship between the proteins (Fitch and Smith, 1983)). A systematic framework where alignments can be rigorously compared (with probabilistic support) and allow users to differentiate between competing alignments, is lacking.

Further, as noted in §2.3, capturing relationships of the *twilight zone* pairs of protein sequences continues to be a challenging problem in the current state of the art. Recent sequence aligners attempt to address this difficulty by incorporating heuristics/additional sources of information (Do et al., 2006; Larkin, 2007; Edgar, 2004; Al Ait et al., 2013), or even allowing partial user-intervention to define reference (anchor) points as a guide to alignable regions (Al Ait et al., 2013), or to manually calibrate gap placements (Chang and Benner, 2004). Yet it requires some extra manual effort by a user to come up with a sensible sequence alignment.

This thesis is motivated to address the aforementioned shortcomings

This thesis is an attempt to rectify the limitations discussed above. Specifically, Chapter 4 lays the foundations of a statistical framework to address the protein sequence alignment problem under the Bayesian and information-theoretic criterion of Minimum Message Length. The framework provides useful statistical properties for unsupervised inference of alignments, without making user-specified choices, and by systematically handling the *complexity* versus *fit* trade-off. The thesis further shows how competing alignments can be explored and visualised. Chapter 5 extends this framework by utilising divergence-time-dependent substitutions and gaps under an existing substitution model, to address the disconnect in their treatment. Chapter 6 presents methods to fully learn new models of amino acid substitution and associated gaps from any given benchmark of protein alignments.

Chapter 3

Statistical Inference with Information Theory

"*The best explanation of the facts is the shortest*"

– Chris S. Wallace

This chapter presents the statistical foundations that underpin the research in this thesis. It covers an introduction to probabilistic modelling, Information theory and a discussion on the general method of statistical inductive inference using the Minimum Message Length (MML) criterion. All concepts covered here provide building blocks for the design and development of the proposed MML framework, to model and address the protein sequence alignment problem described in the succeeding chapters.

3.1 Primer on Probability

3.1.1 Basics

The probability Pr(.) of an outcome is a measure of its *uncertainty*. The space of outcomes Ω can be either discrete or continuous. Accordingly, the associated random variable X over the outcomes is either discrete or continuous. For a discrete sample space, a probability mass function (PMF) conveys the probability for each value of X. The law of probability ensures that $\sum_{\forall x_i \in \Omega} \Pr(X = x_i) = 1$. The expected value of X, denoted as $\mathbb{E}(X)$, to be observed on average under a PMF is given by $\sum_{x_i \in \Omega} x_i \cdot \Pr(X = x_i)$.

On the other hand, for a continuous random variable X, a probability density function (PDF) $f(X)$ explains how the relative-likelihood varies with X. Unlike PMF, PDF does not define an absolute probability for a single X value.[1] Using a PDF, the probability that X falls within a certain range is the area under the function within the range. Again, $\int_{\forall x \in \Omega} f(X = x) = 1$. For real valued sample space, the expectation $\mathbb{E}(X) = \int_{\forall x \in \Omega} x \cdot f(X = x)$.

Many different probability models have been recognised and formulated to describe various events and their outcomes. For instance, the binomial distribution is a discrete probability model that can explain the number of 'successes' of an experiment with independent binary

[1] The absolute likelihood of a continuous value is considered 0, as the precision at a single point is infinite.

outcomes (e.g. coin toss). The geometric distribution models the number of consecutive independent trials needed until a certain success outcome is observed. As for continuous probability distributions, a well known example is the Normal (Gaussian) distribution which is used to model many natural processes. In addition, exponential distribution families (such as the previously encountered Gumbel distribution in §2.4.2) and Dirichlet distribution are widely used for diverse modeling purposes. A probability model may have different statistical parameters to specify aspects such as its overall shape, variance, location and skewness. For example, the normal distribution has mean and standard deviation parameters which control its bell shaped curve.

Probability theory describes the following well-known laws.

Suppose a joint occurrence of two events denoted by two random variables X and Y, producing $X = x_i$ and $Y = y_j$ outcomes. Their *joint probability* is defined by the *product rule* as,

$$\begin{aligned} \Pr(X = x_i, Y = y_j) &= \Pr(X = x_i) \cdot \Pr(Y = y_j \mid X = x_i) && (3.1) \\ &= \Pr(Y = y_j) \cdot \Pr(X = x_i \mid Y = y_j) && (3.2) \end{aligned}$$

Here, $\Pr(X = x_i \mid Y = y_j)$ or $\Pr(Y = y_j \mid X = x_i)$ denote the respective *conditional probability* (the occurrence probability of one event given the occurrence of the other). If the two events are independent, this converts simply into the product of individual occurrence probabilities as:

$$\Pr(X = x_i, Y = y_j) = \Pr(X = x_i) \cdot \Pr(Y = y_j)$$

The "total law of probability" gives the *total* or *marginal* probability of any event ($\Pr(X = x_i)$ or $\Pr(Y = y_j)$) as the summation (if discrete variable) or integration (if continuous variable) over all possible outcomes of the other event:

$$\begin{aligned} \Pr(Y = y_j) &= \sum_{\forall x_i \in \Omega_X} \Pr(X = x_i) \cdot \Pr(Y = y_j \mid X = x_i) && (3.3) \\ \Pr(X = x_i) &= \sum_{\forall y_j \in \Omega_Y} \Pr(Y = y_j) \cdot \Pr(X = x_i \mid Y = y_j) \end{aligned}$$

.

3.1.2 Bayes Theorem

Following the above fundamental rules of probability, Bayes (1763) presented the below formula as the Bayes theorem:

$$\Pr(X = x_i \mid Y = y_j) = \frac{\Pr(X = x_i, Y = y_j)}{\Pr(Y = y_j)} \tag{3.4}$$

illustrating the relationship between *conditional*, *joint* and *marginal* probabilities.

Bayes theorem forms the backbone for Bayesian inference. It gives a *degree of belief* interpretation to the probability of an event based on evidence.

Applying Equations 3.1 and 3.3 to the above, we get:

$$\underbrace{\Pr(X = x_i \mid Y = y_j)}_{\text{Posterior}} = \frac{\overbrace{\Pr(X = x_i)}^{\text{Prior}} \cdot \overbrace{\Pr(Y = y_j \mid X = x_i)}^{\text{Likelihood}}}{\underbrace{\sum_{\forall x_i \in \Omega_X} \Pr(X = x_i) \cdot \Pr(Y = y_j \mid X = x_i)}_{\text{Marginal}}} \tag{3.5}$$

The term *prior* reflects the amount of belief we put into the occurrence of an event before its occurs. Prior probabilities are commonly modelled using prior evidence. As an example, given a long sequence of symbols over a sample space of discrete categorical states (E.g. the amino acid alphabet), the observed frequencies inform the prior probabilities of these symbols (expected to be observed in the future). On the other hand, in the absence of such prior evidence, one might fall back on a bland *uniform prior* probability.

Likelihood informs the likelihood of $Y = y_j$, as the conditional probability of observing its occurrence given the occurrence of $X = x_i$. *Posterior* probability acts as an update to the prior belief $\Pr(X = x_i)$ upon seeing new evidence of it in terms of $Y = y_j$. In other words, it is the probability of $Y = y_j$ given $X = x_j$. When the *prior* probability distribution and *posterior* probability distribution belongs to the same family of probability distributions, they are called *conjugate* probability distributions.

3.2 Primer on Information Theory

3.2.1 Information and Encoding over a Message

Information is what reduces our uncertainty (Wallace, 2005). Intuitively, the higher the probability of some event, the lesser the amount of information required to communicate that event lossslessly, and vice versa. Conceptually, information is a communication between a *transmitter* and a *receiver*, in the form of a message. A *message* can convey any number of events, encoded using a dictionary that maps each possible event to a *codeword*. Thus a message can be viewed as a concatenated string of codewords encoding a given set of data.

The communication ensures a *lossless transmission* of the message. This means the encoded message is completely reversible to extract original data by the receiver through a process of decoding. The details of encoding-decoding is based on a *codebook* that includes general things such as the codeword dictionary and agreed conventions of communication etc. For example, a DNA string `ACCGAATAAA` can be encoded using a fixed length coding scheme: `<A = 00, C = 01, G = 10, T = 11>` with 2-bit codewords. Here, the receiver exactly knows to extract codewords every 2-bits for decoding.

3.2.2 Information Content and Entropy

In 1948, Claude Shannon in his seminal work "*A Mathematical Theory of Communication*" (Shannon, 1948) established the formal principles of optimal lossless data transmission. He defined a measure of information content $I(E)$ for any event E, based on the event's uncertainty:

$$I(E) = -\log_2(\Pr(E)) \quad \text{bits} \tag{3.6}$$

Also known as *Shannon information*, this provides a lower bound to the optimal codeword length required to losslessly encode this event in a message, under the probability model given

by Pr(·). Note that the base of the logarithm defines the unit of measurement.[2] (All Information measures throughout this thesis will be considered in bits, unless stated otherwise).

Further, *Shannon entropy* H measures the expected codeword length per event under the encoding scheme as follow:

$$H = \sum_{\forall E \in \Omega} \Pr(\mathrm{E}) I(E) \quad \text{bits per event} \tag{3.7}$$

Entropy can be a basis for comparing two encoding systems as it informs how economical a coding scheme is.

3.2.3 Kullback-Leibler Divergence

Kullback-Leibler (KL) divergence, denoted by D_{KL}, informs a measure of *relative* entropy needed for encoding data over a probability model $p(\cdot)$ relative to another probability model $q(\cdot)$.

$$D_{KL}[p, q] = \sum_{\forall E \in \Omega} p(E) \log_2 \left(\frac{p(E)}{q(E)} \right) \quad \text{bits per event} \tag{3.8}$$

where $D_{KL}[p, q] \geq 0$. This divergence measure is useful for evaluating one probability model against another. However it is not a strict distance *metric* since it is non-symmetric ($D_{KL}[p, q] \neq D_{KL}[q, p]$) and does not always satisfy the triangle inequality. We do have $D_{KL}[p, q] = 0$ if and only if $p = q$.

3.2.4 Fixed-length Codes, Variable-length Codes and Adaptive Codes

From Equation 3.6, a fixed-length coding scheme results from a uniform probability distribution over the sample space. For example, 2-bit codewords in the previous example dealing with $E \in \{\texttt{A,C,T,G}\}$ all yield a uniform probability of $\frac{1}{2^2} = 0.25$ probability, as it assumes that the true distribution that generate these symbols is uniform. When this assumption does not hold, a variable-length coding scheme is more efficient (i.e. concise). The variable-length coding scheme requires a more complex method (compared to fixed-length coding scheme) of decoding the codes correctly. A common method in data compression is to ensure one codeword is not a prefix of another. Such scheme is known as *prefix-free* coding system.

Huffman coding (Huffman, 1952) is a prefix-free code system where all unique paths from a root to the leaves in a binary tree constructed based on the probabilities, account for variable length binary codes of each symbol. It ensures that more probable symbols are assigned shorter codewords, and those that are less probable get assigned longer ones.

Adaptive encoding is an approach for dynamically updating the coding scheme, that is useful for estimating the message lengths of multi-state strings. Initially, both sender and receiver agree on a prior probability model for each state in the state space. As the transmission proceeds with each successful transmission of a single event, both parties update their probability models using a set of counters maintained to keep track of the number of times each possible event has been observed so far (Cleary and Witten, 1984). The adaptive encoding scheme works as follow: the sender encodes the first symbol of the sequence with an initially agreed probability, while updating the relevant counter. The receiver can decode this symbol using the agreed

[2] Base-2 logarithm gives the measure in *bits*, base-10 in *dits*, base-e gives *nits* etc.

Table 3.1: How adaptive encoding works: At time $t = 0$, the counters are set to 1 (representing an initial pseudocount for each possible state in the state space Ω). As the time passes, at each discrete time point t, an observation of some state $s_x \in \Omega$ is made at some $t = i$. Each state $s_x \in \Omega$ has the counter c_x which is always updated upon its observation at some time t. At $t = N$, we have N total number of state observations (i.e. the length of the state sequence observed so far is N). Accordingly, at any time $t = i$, the encoding probability of state s_x is: $\left(\frac{n_{s_x}}{k+i}\right)$, where n_{s_x} is the number of s_x state observations so far.

	At time $t = 0$	$t = 1$	$t = 2$	$t = 3$		$t = N$
		$\mathbf{o_1}$	$\mathbf{o_2}$	$\mathbf{o_3}$		$\mathbf{o_N}$
Counters	**Pseudocount**	$\mathbf{s_2}$	$\mathbf{s_1}$	$\mathbf{s_2}$	. . .	$\mathbf{s_3}$
c_1	1	1	2	2	. . .	$n_{s_1} + 1$
c_2	1	2	2	3	. . .	$n_{s_2} + 1$
c_3	1	1	1	1	. . .	$n_{s_3} + 1$
.	.	.	.	.	. . .	.
.	.	.	.	.	. . .	.
.	.	.	.	.	. . .	.
c_k	1	1	1	1	. . .	$n_{s_k} + 1$
Count sum	k	$k + 1$	$k + 2$	$k + 3$	. . .	$k + N$

model and then update the counter. Now both sides have new probabilities to continue the encoding and decoding process.

For instance, suppose a general multi-state model with a state space $\Omega = \{s_1, s_2, ...s_k\}$ of size k is applied to encode a sequence $S = (s_2 s_1 s_2 \dots s_3)$. Table 3.1 illustrates how dynamic coding works with counters $\{c_1, c_2, ...c_k\}$ for all k states, initially setting to 1 (i.e. uniform model) as the agreed model. A column o_i gives the counter update upon seeing i^{th} state observation in the sequence. There, any n_j refers to the number of j^{th} state observations so far, and N is the total number of observations so far.

The total message length of S under adaptive encoding is given by:

$$I(S) = -\log_2 \left(\frac{(n_{s_1})!(n_{s_2})!...(n_{s_k})!}{\left(\frac{(k+N-1)!}{(k-1)!}\right)} \right) \quad \text{bits} \tag{3.9}$$

3.2.5 Variable-length Code for Integers

This section discusses variable-length encoding of any integer $n \in \mathbb{Z}$, specifically Wallace tree codes (Wallace and Patrick, 1993). As seen in later chapters, the research in this thesis utilises these variable-length integer codes.

An integer coding scheme becomes *universal* when the true probability distribution satisfies $\Pr(n) \geq \Pr(n+1)$ and the `|codeword(n)|` $< -c. \log[\Pr(n)]$ (within a linear scaling of the expected optimal code length) for some constant c (Elias, 1975).

Wallace and Patrick (1993) proposed a universal, prefix free integer encoding scheme by mapping integers to the countable set of *strict* binary trees (where each internal node has exactly two children). The number of possible strict binary trees over k internal nodes is given by the Catalan number C_n:

$$C_k = \frac{1}{k+1} \binom{2k}{k}$$

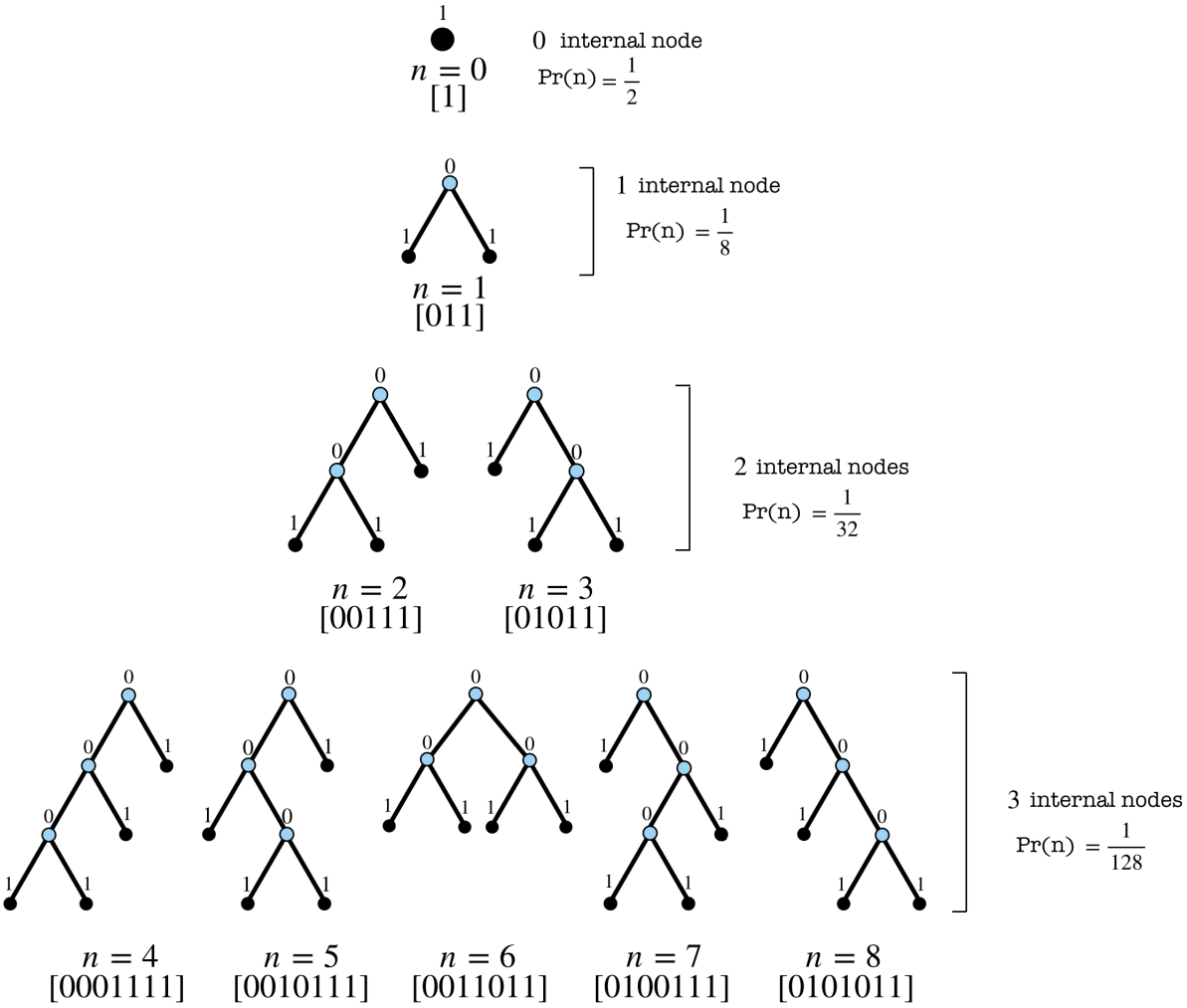


Figure 3.1: Wallace tree code construction for integers(Wallace and Patrick, 1993) (Note: Order of tree permutation construction gives priority from left to right)

Thus, the set of positive integers $n = 1, 2, 3 \ldots$ can be mapped to the the set of strict binary trees with progressively increasing number ($k = 0, 1, 2, \ldots$) of internal nodes. For any fixed k, there are C_k number of positive integers distinctly mapped to corresponding strict binary trees with k internal nodes. The codeword for the mapped integer is simply the pre-order traversal of its corresponding strict binary tree, where the internal nodes denote 0 and leaf nodes denote 1 in the traversal. Thus, if the corresponding binary tree has k internal nodes, the mapped codeword takes $2k + 1$ bits in lengths to represent. Thus, from this process, a perfectly decodable variable-length integer codes can be derived with attractive asymptotic properties (Allison et al., 2019).

Although, the resultant set of variable-length integer codes from this mapping is over positive integers, they can be mapped to a non-negative integer set by shifting the code-assignments to start from $n = 0$ instead of $n = 1$ (see Figure 3.1). Further, the positive integer codes can be mapped to any countable set, for example, the set of all integers ordered as $\{0, 1, -1, 2, -2, 3, -3, \ldots\}$.

3.3 Minimum Message Length Paradigm for Model Selection

The Minimum Message Length principle (MML)(Wallace and Boulton, 1968; Wallace, 2005) lays the groundwork for statistical inductive inference with Information theory. It provides a Bayesian framework for model comparison, selection and parameter estimation.

MML is best understood as a communication of a two-part message between a hypothetical sender and receiver. A sender encodes and transmits a hypothesis (model) H (in the first part) and then some observed data D under the assumption that H is true (in the second part). This communication is based on a properly defined lossless communication protocol over a set of agreed rules (the *codebook*). The *codebook* also contains common knowledge and any preconceived notions regarding the data being transmitted. This ensures a message composition that encompasses all information necessary for its complete decodability at the receiver end.

This framework combines Information theory with Bayesian statistics. It reformulates the Bayes theorem (Bayes, 1763) in information-theoretic terms (Shannon, 1948). Building on the introduction to Bayes theorem in §3.1.2, the joint probability of any hypothesis H explaining the data D can be defined using the product rule as:

$$\Pr(H,D) = \Pr(H)\Pr(D\ |H)$$

Applying logarithm on both sides of the above rule, and reinterpreting it in terms of the Shannon information content defined in §3.2, this rule can be seen as the lossless encoding length of communicating the hypothesis H and data D given H:

$$I(H,D) = \underbrace{I(H)}_{\text{First part length}} + \underbrace{I(D|H)}_{\text{Second part length}} \tag{3.10}$$

In this framework the best and the optimal hypothesis (model) H^* is the one that minimises $I(H,D)$ as the objective function. This formulation therefore allows comparing any two competing hypotheses. Given two hypotheses H_1 and H_2, the difference in message lengths

$$\Delta I = I(H_1,D) - I(H_2,D)$$

gives their posterior log-odds ratio:

$$\Delta I = -\log_2(\Pr(H_1,D)) + \log_2(\Pr(H_2,D)) = \log_2\left(\frac{\Pr(D)\Pr(H_2\ |D)}{\Pr(D)\Pr(H_1\ |D)}\right) = \log_2\left(\frac{\Pr(H_2\ |D)}{\Pr(H_1\ |D)}\right) \quad \text{bits}$$

If $\Delta I > 0$, this indicates that H_2 is $2^{\Delta I}$ more likely than H_1, otherwise vice versa.

MML provides a natural statistical test of significance:

$$\begin{aligned} I_{\text{NULL}}(D) - I(H,D) &> 0 \implies \text{Accept the hypothesis} \\ I_{\text{NULL}}(D) - I(H,D) &\leq 0 \implies \text{Reject the hypothesis} \end{aligned}$$

That is, any hypothesis H whose encoding length $I(H,D)$ is lesser than a *null model* message length (denoted by $I_{\text{NULL}}(D)$) is accepted; otherwise rejected. The null model implies a raw (i.e. *as is*) encoding of data D without the support of any "interesting" hypothesis.

The MML paradigm includes a trade-off between hypothesis *complexity* and its *fit* with the data. When H is a model with parameters, the first part of the message contains statements of all of its associated parameter values (to an optimal precision that is resolved under the MML criterion). The statement length of the hypothesis, $I(H)$, captures this complexity. The

higher the complexity is, the longer $I(H)$ becomes. The fidelity of the hypothesis to explain the observed data is captured by the length of the second part, $I(D|H)$. The shorter the $I(D|H)$ is, the better it is at explaining the data. Hence, MML achieves a good, optimal trade-off between both hypothesis complexity and fit.

3.3.1 Single Continuous Parameter Estimation using MML

We will now explore the mathematical details of the MML technique of Wallace and Freeman (1987) to estimate parameters, known as MML87, which forms the backbone of parameter estimation in this thesis. Specifically, this section will explore the MML87 technique first for the single continuous parameter case, before generalising it in the next section.

Let θ be a continuous parameter of a statistical model, with a prior distribution $h(\theta)$ as shown in Figure 3.2a. Also let the model explain some observed data D with a likelihood function $L(\theta)$. Under the MML inference framework, search for the optimal parameter value for θ is formulated as optimising a two-part message length explained in the previous section.

The first part of the message must encode the real valued θ parameter itself. Any real valued parameter can only be stated to a limited precision, which must be chosen such that the associated total two-part message is minimised. Suppose the precision of parameter value is denoted by A_θ. Figure 3.2b illustrates the corresponding continuous range $[\theta - \frac{A_\theta}{2}, \theta + \frac{A_\theta}{2}]$ of uncertainty for θ under MML87. The smaller this region is, the more precise is θ. Under the MML87 assumption that prior changes slowly in the neighbourhood of any value in the x axis, the probability of the parameter θ can be estimated as:

$$\Pr(\theta) = h(\theta)A_\theta$$

Accordingly, the length of the first part message becomes $-\log_2 [h(\theta)A_\theta]$.

The second part of the message encodes the data D given the value of θ. As a result, the message length $I(D|\theta)$ is simply given by the negative log likelihood function $\mathcal{L}(\theta) = -\log_2 [L(\theta)]$.

Note that the decoder only sees an uncertain region of θ defined with the precision A_θ. For any value around θ within this region, a corresponding likelihood value exists. Thus, an expected message length over all those possible values is computed for $I(D|\theta)$:

$$I(D|\theta) = \frac{1}{A_\theta} \int_{x=-\frac{A_\theta}{2}}^{x=+\frac{A_\theta}{2}} \mathcal{L}(\theta + x)dx$$

MML87 applies the Taylor series expansion to $\mathcal{L}(\theta + x)$ as:

$$\mathcal{L}(\theta + x) = \mathcal{L}(\theta) + \mathcal{L}'(\theta)x + \mathcal{L}''(\theta)\frac{x^2}{2} + O(x^3) \text{ terms}$$

Truncating the series after the quadratic (x^2) term, and applying the integral in the neighbourhood of $x = \theta \pm \frac{A_\theta}{2}$, results in:

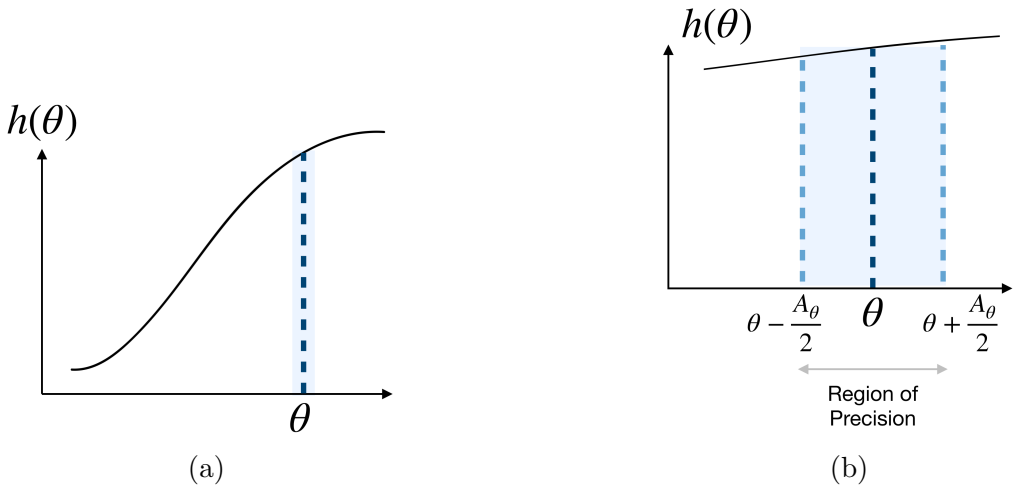
$$I(D|\theta) = \mathcal{L}(\theta) + \mathcal{L}''(\theta)\frac{A_\theta^2}{24}$$

where $\mathcal{L}''(\theta)$ is the second derivative of the negative log likelihood function. It is also named as the empirical Fisher or the Hessian. This figure informs the sensitivity of the negative log likelihood to the variations in θ (Wallace and Freeman, 1987). A high $\mathcal{L}''(\theta)$ value implies a sharp minimum of $\mathcal{L}(\theta)$. This means, even a slight variation of θ can increase $I(D|\theta)$ quite a

lot and thus, in such case we prefer a much higher precision for θ. On the other hand, when the Fisher is small, the radius of the curvature at θ will be larger, indicating less variability in $I(D|\theta)$ for precision errors in θ. Therefore slight increases in $\mathcal{L}(\theta)$ can be tolerated with somewhat a less precise statement of θ. For instance, see Figure 3.3.

Consequently, the total message length $I(\theta, D)$ is given by:

$$I(\theta, D) = -\log_2 [h(\theta)A_\theta] + \mathcal{L}(\theta) + \mathcal{L}''(\theta)\frac{A_\theta^2}{24}$$



(a) (b)

Figure 3.2: Prior distribution $h(\theta)$ for the continuous parameter θ

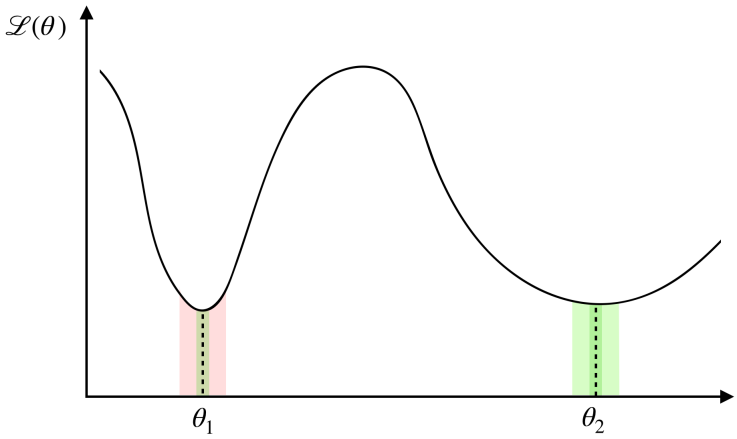


Figure 3.3: This illustrates how the precision of θ affects the computation of expected $I(D|\theta)$. Consider θ_1 and θ_2 as parameter values reside against a steep valley and flat valley of the negative log likelihood function, respectively. For θ_1, a lower precision (highlighted by the red region) is less tolerable as it accounts for a higher variation in $I(D|\theta)$. On the other hand, a higher precision for θ_1 (highlighted by the green region) works well for computing an accurate expected value of $I(D|\theta_1)$, as $I(D|\theta)$ does not vary much within that region. In contrast, θ_2 can tolerate both lower precision and higher precision for θ, as for both regions, $I(D|\theta)$ does not vary significantly. (Note: Red color refers to less tolerance; Green color refers to more tolerance)

This acts as an objective function to choose the optimal θ parameter and its optimal precision (A_θ). By solving $\frac{d}{dA_\theta} I(\theta, D) = 0$, it turns out that $A_\theta = \sqrt{\frac{12}{\mathcal{L}''(\theta)}}$, which is dependent on the empirical Fisher (observed over data D). However the hypothetical receiver is unaware of the optimal A_θ unless it is encoded within the message. This is impossible since A_θ, as a real value, should again be stated upto a certain optimal precision. It leads to an *infinite regress* of precision statements (Oliver and Hand, 1994). The solution is to use the *expected* Fisher instead the observed Fisher, which takes the expectation of $\mathcal{L}''(\theta)$ over all possible datasets. By defining the expected Fisher:

$$Fisher(\theta) = \int L(\theta) \cdot \mathcal{L}''(\theta) dD$$

the optimal A_θ is found to be $\sqrt{\frac{12}{Fisher(\theta)}}$. Substituting that into the total message length expression, we get:

$$I(\theta, D) = \frac{1}{2}\log_2\left(\frac{1}{12}\right) + \frac{1}{2}\log_2\left[Fisher(\theta)\right] - \log_2\left[h(\theta)\right] + \mathcal{L}(\theta) + \frac{1}{2}$$

Note that, common non-Bayesian methods such as maximum likelihood estimation do not take into account the optimal precision of parameters nor the complexity involved with every parameter statement, compared to MML which formulates an honest representation of all parameters and their complexities.

3.3.2 Multiple Continuous Parameter Estimation using MML

We can generalise the single continuous parameter estimation to higher dimensions, for estimating a continuous parameter vector of the form $\vec{\theta} = [\theta_1, \theta_2, ...\theta_n]$. Define prior $h(\vec{\theta})$, likelihood $L(\vec{\theta}|D)$ and negative log likelihood $\mathcal{L}(\vec{\theta}) = -\log_2\left(L(\vec{\theta}|D)\right)$. For the single-parameter case, the probability of θ within the uncertainty region was simply the associated area under the $h(\theta)$ curve. For $\vec{\theta}$, this is a hypervolume V. Let the precision of $\vec{\theta}$ be V_θ

The equivalent total message length is:

$$I(D, \vec{\theta}) = I(\vec{\theta}) + I(D|\vec{\theta}) = -\log_2\left(h(\vec{\theta})V_\theta\right) + I(D|\vec{\theta}) \tag{3.11}$$

Similar to the single-parameter case, we compute $I(D|\vec{\theta})$ as the expected message length over all possible vectors falling in the region of uncertainty around $\vec{\theta}$. It is then expanded through the multi-variate Taylor series, and truncated after the quadratic term:

$$\begin{aligned} I(D|\vec{\theta}) &= \frac{1}{V_\theta}\int_{V_\theta} \mathcal{L}(\vec{\theta} + \vec{x}) d\vec{x} \\ &\approx \frac{1}{V_\theta}\int_{V_\theta} \left(\mathcal{L}(\vec{\theta}) + \mathcal{L}'(\vec{\theta})\vec{x} + \frac{\vec{x}^T\mathcal{L}''(\vec{\theta})\vec{x}}{2!}\right) d\vec{x} \\ &\approx \mathcal{L}(\vec{\theta}) + \frac{1}{2V_\theta}\int_{V_\theta} \vec{x}^T\mathcal{L}''(\vec{\theta})\vec{x} \quad d\vec{x} \end{aligned}$$

Again, since $\mathcal{L}''(\vec{\theta})$ is the observed Fisher matrix, it is replaced by the expected Fisher matrix $Fisher(\vec{\theta})$.

$$I(D|\vec{\theta}) = \mathcal{L}(\vec{\theta}) + \frac{1}{2V_\theta}\int_{V_\theta} \vec{x}^T Fisher(\vec{\theta})\vec{x} \quad d\vec{x}$$

$Fisher(\vec{\theta})$ is a square symmetric matrix with an eigen decomposition of $S\Lambda S^{-1}$, where Λ is a diagonal matrix with eigenvalues and S is an orthogonal matrix of eigenvectors. The orthogonality ensures $S^{-1} = S^T$. Thus, the integral part of the above expression can be simplified using the following non-linear transformation:

$$\begin{aligned} \vec{x}^T Fisher(\vec{\theta})\vec{x} &= \vec{x}^T S\Lambda S^T \vec{x} \\ &= Z^T \Lambda Z \\ &= \underbrace{Z^T\sqrt{\Lambda}}_{\vec{y}^T}\underbrace{\sqrt{\Lambda}Z}_{\vec{y}} \\ &= \vec{y}^T\vec{y} \end{aligned}$$

where $Z = S^T$. This permits a spherical reparameterisation that transforms V_θ into a new hypervolume U_θ, with $\vec{y} = \sqrt{\Lambda}S^T\vec{x}$. Therefore, the associated differential volume $\frac{U_\theta}{V_\theta}$ is given by the determinant of the Jacobian matrix $\frac{d\vec{y}}{d\vec{x}} = det[\sqrt{\Lambda}S^T]$. This expression can be further simplified as follows:

$$\begin{aligned} det[\sqrt{\Lambda}S^T] &= det[\sqrt{\Lambda}][det[S^T] \\ &= det[\sqrt{\Lambda}] \end{aligned}$$

due to the determinant of an orthogonal matrix being 1. This again implies:

$$det[\sqrt{\Lambda}S^T] = \sqrt{det[\Lambda]}$$

Further,

$$\begin{aligned} det[Fisher(\theta)] &= det[\mathcal{L}^{''}(\vec{\theta})] \\ &= det[S\Lambda S^T] \\ &= det[\Lambda] \\ \implies det[\sqrt{\Lambda}S^T] &= \sqrt{det[Fisher(\vec{\theta})]} \end{aligned}$$

Consequently, the original prior $h(\vec{\theta})$ is transformed into some $g(\vec{\phi})$ with,

$$\begin{aligned} g(\vec{\phi})U_\theta &= h(\vec{\theta})V_\theta \\ \implies g(\vec{\phi}) &= h(\vec{\theta})\frac{V_\theta}{U_\theta} \\ \implies g(\vec{\phi}) &= \frac{h(\vec{\theta})}{\sqrt{det[Fisher(\vec{\theta})]}} \end{aligned} \tag{3.12}$$

When the total message length is converted to this new parameterisation, we get:

$$I(\theta, D) = -\log_2[g(\phi)U_\theta] + \mathcal{L}(\vec{\phi}) + \frac{1}{2U_\theta}\int_{U_\theta} y^T y \quad du$$

where $\mathbb{E}(y^T y) = \frac{1}{U_\theta}\int_{U_\theta} y^T y \quad du$ is the expected value of $y^T y$.

From Conway and Sloane (1984), $\mathbb{E}(y^T y)$ is given by $d\kappa_d U^{2/d}$ where d is the number of free dimensions and κ_d is the Conway constant (for optimal quantising lattice packing) for d

dimensions. Known constants are $\kappa_1 = \frac{1}{12}; \kappa_2 = \frac{5}{36\sqrt{3}}$ $\kappa_3 = \frac{19}{192(2^{\frac{1}{3}})}$. Wallace (2005) provides a lower and upper bound for computing the lattice constant for higher dimensions (in this thesis, the upper bound is used for $d > 3$):

$$\frac{\{(d/2)!\}^{2/d}}{(d+2)\pi} < \kappa_d < \frac{\{(d/2)!\}^{2/d}(2/d)!}{\pi d} \tag{3.13}$$

As d increases, both lower and upper bounds approach the same value (See Table 3.2).

The optimal U_θ is given by $\kappa_d^{-\frac{d}{2}}$ upon solving $\frac{\partial}{\partial U_\theta} I(\theta, D) = 0$. Substituting this optimal value gives:

$$I(\theta, D) = -\log_2 [g(\vec{\phi})] + \frac{d}{2}\log_2 (\kappa_d) + \mathcal{L}(\vec{\phi}) + \frac{d}{2}$$

Converting back $g(\vec{\phi})$ to the original prior $h(\vec{\theta})$ using Equation 3.12 results in:

$$I(\vec{\theta}, D) = \underbrace{\frac{d}{2}\log_2 (\kappa_d) - \log_2 [h(\vec{\theta})] + \frac{1}{2}\log_2 [det[Fisher(\vec{\theta})]]}_{\text{First part: } I(\theta)} + \underbrace{\mathcal{L}(\vec{\theta}) + \frac{d}{2}}_{\text{Second part: } I(D|\theta)} \tag{3.14}$$

Here, $\frac{\kappa_d^{-\frac{d}{2}}}{\sqrt{det[Fisher(\vec{\theta})]}}$ is the volume of the region of uncertainty for $\vec{\theta}$. $\frac{d}{2}$ in the second part accounts for the expected correction of rounding off $\vec{\theta}$ to a point estimate $\hat{\vec{\theta}}$.

The MML estimate $\hat{\vec{\theta}}_{MML}$ is the $\vec{\theta}$ that minimises $I(\vec{\theta}, D)$.

$$\hat{\vec{\theta}}_{MML} = \underset{\forall \vec{\theta} \in \Omega_{\vec{\theta}}}{\operatorname{argmin}} I(\vec{\theta}, D)$$

This method requires a sufficient amount of data for successful parameter estimation, necessarily a significant number of data points more than the number of free parameters to be estimated. Otherwise, the first part may result in a negative length, due to the approximation of $\Pr(\vec{\theta})$. The assertion of $\Pr(\vec{\theta}) = h(\vec{\theta}) \times V_\theta$ does not hold in such case, leading to a case of $\Pr(\vec{\theta}) >$

Table 3.2: Lower and upper bounds of quantising lattice constant (Conway and Sloane, 1984) for various numbers of free dimensions

	Lower bound		**Upper bound**	
d	κ_d	$\log_2(\kappa_d)$	κ_d	$\log_2(\kappa_d)$
1	0.0833333	-3.58496	0.5	-1
2	0.0795775	-3.6515	0.159155	-2.6515
3	0.0769669	-3.69962	0.115803	-3.11026
10	0.0691043	-3.85508	0.0761393	-3.71521
19	0.0657551	-3.92675	0.0689884	-3.8575
20	0.0655245	-3.93182	0.0685705	-3.86627
24	0.0647498	-3.94898	0.0672195	-3.89498
25	0.0645858	-3.95264	0.0669434	-3.90091
50	0.0622978	-4.00468	0.0633927	-3.97954
100	0.0608011	-4.03976	0.0613253	-4.02737

1. Therefore, to handle this, Wallace (2005) suggested the following approximation, which continues to remain invariant under the nonlinear transformations of $\vec{\theta}$:

$$I(\vec{\theta}) \approx \frac{1}{2}\log_2\left(1 + \frac{det[Fisher(\vec{\theta})]\kappa_d^d}{[h(\vec{\theta})]^2}\right) \tag{3.15}$$

3.3.3 MML estimation of Parameters for a Multi-state Distribution

Here, we go through the MML estimation procedure applied to the multi-state probability distribution. Multi-state models are heavily used in this thesis for modelling protein sequences and their alignments.

A multi-state probability distribution describes a string generated over a fixed alphabet $\Omega = \{s_1, \ldots, s_{|\Omega|}\}$ with $k = |\Omega|$ symbols. A k-dimensional probability vector $\vec{\theta} = \{\Pr(s_1), \ldots, \Pr(s_{|\Omega|})\}$ parameterises the model, with a probability value for each state. This vector is $\mathbb{L}1$-normalised (i.e. $\sum_{i=1}^{k} \Pr(s_i) = 1$). The standard unit $k-1$ simplex is the set of all possible k-dimensional $\mathbb{L}1$-normalised vectors (Allison, 2018). Thus, $\vec{\theta}$ is a point in a $k-1$ unit simplex with $k-1$ free parameters. This has a hypervolume of $\frac{\sqrt{k}}{(k-1)!}$. Figure 3.4 shows a unit 1-simplex and unit 2-simplex.

The goal here is to estimate the total message length $I(\vec{\theta}, D)$ given by Equation 3.14 specific to the multi-state model, and infer $\vec{\theta}$ that minimises it, using the method of MML87.

Let $\vec{\theta} = [\theta_1, \theta_2, \ldots \theta_k]$ define the probability vector where θ_k is the probability of k^{th} state. The model has $(k-1)$ degrees of freedom (free parameters), as the following holds for any θ_k:

$$\theta_k = 1 - \sum_{i=1}^{k-1} \theta_i \tag{3.16}$$

Suppose data D at hand as a sequence of states over Ω, and the corresponding frequency (observed count) vector for each state as: $\{x_1, x_2, \ldots x_k\}$. If the total number of observations

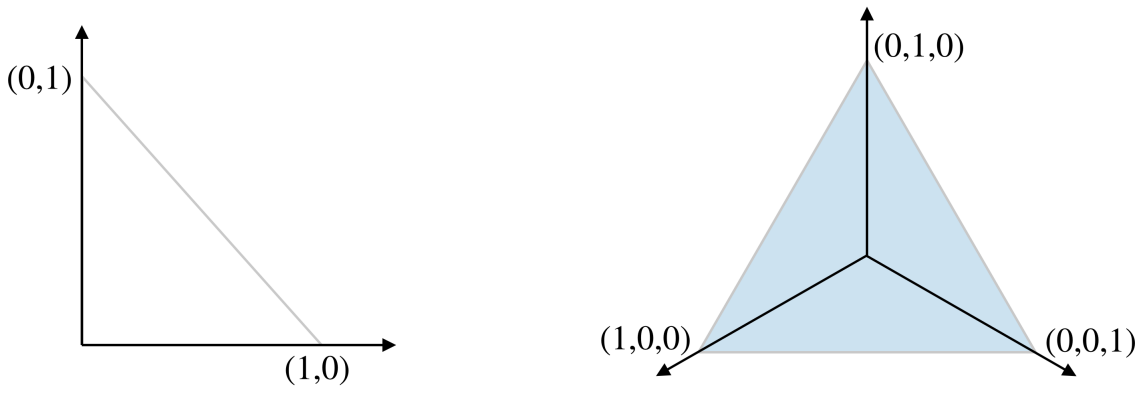


Figure 3.4: (a) unit 1-simplex visualisation (describing any 2-dimensional probability vector represented by a point in 1-dimensional line from (0,1) to (1,0)), and (b) unit 2-simplex visualisation (describing any 3-dimensional probability vector represented by a point in the light blue highlighted surface defined by (0,1,0), (0,0,1) and (1,0,0))

(data) is N, then:

$$\sum_{i=1}^{k} x_i = N \tag{3.17}$$

Assuming independent and identically distributed data points in D, the negative log likelihood $\mathcal{L}(\vec{\theta})$ can be computed using the likelihood $f(\vec{\theta}|D)$ as follows:

$$\begin{aligned} \mathcal{L}(\vec{\theta}) &= -\log_2 \left[f(\vec{\theta}|D)\right] && (3.18)\\ &= -\log_2 \left(\prod_{i=1}^{k} \theta_i^{x_i}\right) && (3.19)\\ &= -\sum_{i=1}^{k} \log_2 \left(\theta_i^{x^i}\right) = -\sum_{i=1}^{k} x_i \log_2 \left(\theta_i\right) && (3.20) \end{aligned}$$

The determinant of the Fisher matrix $det[F(\theta)]$ is obtained through computing the observed second order partial derivatives of $\vec{\theta}$ at first. The first order partial derivatives are given by:

$$\begin{aligned} \frac{\partial}{\partial \theta_i}[\mathcal{L}(\vec{\theta})] &= \frac{\partial}{\partial \theta_i}[-\sum_{i=1}^{k} x_i \log_2 \left(\theta_i\right)] \\ &= \left(-\frac{x_i}{\theta_i} + \frac{x_k}{\theta_k}\right) \end{aligned}$$

Accordingly, we can compute the second order partial derivatives:

$$\begin{aligned} \frac{\partial^2}{\partial \theta_i^2}[\mathcal{L}(\vec{\theta})] &= \left(\frac{x_i}{\theta_i^2} + \frac{x_k}{\theta_k^2}\right) \\ \frac{\partial^2}{\partial \theta_i \partial \theta_j}[\mathcal{L}(\vec{\theta})] &= \frac{x_k}{\theta_k^2} \end{aligned}$$

Since we deal with expected values to avoid *infinite regress*, the expected Fisher is derived using $\mathbb{E}(\frac{\partial^2}{\partial \theta_i^2}[\mathcal{L}(\vec{\theta})])$ and $\mathbb{E}(\frac{\partial^2}{\partial \theta_i \partial \theta_j}[\mathcal{L}(\vec{\theta})])$, by plugging in the expected counts for x_i and x_k:

$$\begin{aligned} \mathbb{E}(\frac{\partial^2}{\partial \theta_i^2}[\mathcal{L}(\vec{\theta})]) &= \frac{N\theta_i}{\theta_i^2} + \frac{N\theta_k}{\theta_k^2} \\ &= \left(\frac{N}{\theta_i} + \frac{N}{\theta_k}\right) \\ \mathbb{E}(\frac{\partial^2}{\partial \theta_i \partial \theta_j}[\mathcal{L}(\vec{\theta})]) &= \frac{N\theta_k}{\theta_k^2} \\ &= \frac{N}{\theta_k} \end{aligned}$$

The determinant of the expected Fisher matrix $Fisher(\theta)$ is:

$$det[Fisher(\vec{\theta})] = \left\| \begin{matrix} \frac{\partial^2 \mathcal{L}}{\partial \theta_1^2} & \frac{\partial^2 \mathcal{L}}{\partial \theta_1 \partial \theta_2} & \frac{\partial^2 \mathcal{L}}{\partial \theta_1 \partial \theta_3} & .. & .. \\ \frac{\partial^2 \mathcal{L}}{\partial \theta_2 \partial \theta_1} & \frac{\partial^2 \mathcal{L}}{\partial \theta_2^2} & \frac{\partial^2 \mathcal{L}}{\partial \theta_2 \partial \theta_3} & .. & .. \\ ... & & ... & & \\ ... & & & ... & \\ ... & & & & \frac{\partial^2 \mathcal{L}}{\partial \theta_{k-1}^2} \end{matrix} \right\| = \left(\frac{N^{k-1}}{\prod_{i=1}^{k} \theta_i} \right) \tag{3.21}$$

(Note: this is a $(k-1) \times (k-1)$ square symmetric matrix)

Defining a prior The simplest prior is to use a distribution $h(\vec{\theta})$ that is uniform. Since $\vec{\theta}$ represents a point in the unit $(k-1)$-simplex, a uniform probability of $\frac{1}{\text{volume(simplex)}}$ can be applied with the previously mentioned volume formula:

$$h(\vec{\theta}) = \frac{(k-1)!}{\sqrt{k}} \tag{3.22}$$

A more flexible prior can be incorporated to support $\vec{\theta}$ estimation with previous knowledge. Dirichlet probability distribution is the conjugate prior for multi-state distribution. It is a family of continuous probability distributions that models the uncertainty of a k-dimensional point in a unit k−1 simplex. Given a Dirichlet model Dir($\vec{\alpha}$) with a parameter vector [α_1,α_2,...α_k] (for α_i>0) that describes a data sample $\vec{x}$. its PDF $f(\vec{x}\ |\vec{\alpha})$ is defined as:

$$f(\vec{x}\ |\vec{\alpha}) = \frac{1}{\mathrm{B}(\vec{\alpha})} \prod_{i=1}^{\mathrm{k}} x_i^{\alpha_i - 1} \tag{3.23}$$

where, B($\vec{\alpha}$) is the multivariate form of the Beta function.

Minimisation of $I(\vec{\theta}, D)$ For obtaining the optimal θ_i using a *uniform prior* $h(\vec{\theta})$, the previously derived statistics from Equation 3.18 and Equation 3.21 are substituted to the total message length $I(\vec{\theta}, D)$ given in Equation 3.14. Solving $\frac{\partial}{\partial \theta_i}[I(\vec{\theta}, D)] = 0$ gives the optimal θ_i estimate as follow.

$$\begin{aligned} \frac{\partial}{\partial \theta_i} \left(\frac{d}{2} \log_2 (\kappa_d) - \log_2 [h(\vec{\theta})] + \frac{1}{2} \log_2 [det[Fisher(\theta)]] + \mathcal{L}(\vec{\theta}) + \frac{d}{2} \right) &= 0 \\ \implies \frac{1}{2} \frac{\partial}{\partial \theta_i} \log_2 [det[Fisher(\vec{\theta})]] + \frac{\partial}{\partial \theta_i} \mathbb{L}(\vec{\theta}) &= 0 \\ \implies \frac{1}{2} \frac{\partial}{\partial \theta_i} \log_2 \left(\frac{N^{k-1}}{\prod_{i=1}^{k} \theta_i} \right) - \frac{\partial}{\partial \theta_i} \sum_{i=1}^{k} x_i \log_2 (\theta_i) &= 0 \end{aligned}$$

This can be further expanded as:

$$-\frac{1}{2} \frac{\partial}{\partial \theta_i} \left(\sum_{i=1}^{k-1} \log_2 (\theta_i) + \log_2 (1 - \sum_{i=1}^{k-1} \theta_i) \right) - \frac{\partial}{\partial \theta_i} \left(\sum_{i=1}^{k-1} x_i \log_2 (\theta_i) + x_k \log_2 (1 - \sum_{i=1}^{k-1} \theta_i) \right) = 0$$

resulting in:

$$\begin{aligned} -\frac{1}{2}\left(\frac{1}{\theta_i}-\frac{1}{\theta_k}\right)-\frac{x_i}{\theta_i}+\frac{x_k}{\theta_k} &= 0 \\ \implies -\frac{x_i+0.5}{\theta_i}+\frac{x_k+0.5}{\theta_k} &= 0 \\ \implies \theta_i &= \frac{\theta_k(x_i+0.5)}{(x_k+0.5)} \end{aligned}$$

To remove θ_k from the estimator, substitute $\sum_{i=1}^{k}\theta_i = 1$, which gives:

$$\sum_{i=1}^{k}\left(\frac{\theta_k(x_i+0.5)}{(x_k+0.5)}\right)=1 \implies \left(\frac{\theta_k}{x_k+0.5}\right)=\left(\frac{1}{N+(\frac{k}{2})}\right)$$

From above, the MML estimate for θ_i under a *uniform prior* can be derived as:

$$\theta_i = \left(\frac{x_i+0.5}{N+\frac{k}{2}}\right) \text{ (Note: that } \hat{\theta}_i > 0 \text{ even when } x_i = 0) \tag{3.24}$$

Separately, a *Dirichlet prior* $\mathrm{Dir}(\vec{\alpha})$ can be applied as $h(\vec{\theta})$. The term $\log_2\left(h(\vec{\theta})\right)$ in Equation 3.11 then becomes:

$$\begin{aligned} \log_2\left[h(\vec{\theta})\right] &= \log_2\left(\frac{1}{B(\vec{\alpha})}\right)+\sum_{i=1}^{k}(\alpha_i-1)\log_2(\theta_i) \\ &= \log_2\left(\frac{1}{B(\vec{\alpha})}\right)+\sum_{i=1}^{k-1}(\alpha_i-1)\log_2(\theta_i)+(\alpha_k-1)\log_2\left(1-\sum_{i=1}^{k-1}\theta_i\right) \end{aligned}$$

Taking the derivative of $\log_2\left[h(\vec{\theta})\right]$, we have:

$$\frac{d}{d\theta_i}\log_2\left[h(\vec{\theta})\right]=\frac{(\alpha_i-1)}{\theta_i}-\frac{(\alpha_k-1)}{\theta_k}$$

Consequently, optimisation routine continues with:

$$\frac{\partial I(\vec{\theta},D)}{\partial\vec{\theta}}=-\frac{(\alpha_i-1)}{\theta_i}+\frac{(\alpha_k-1)}{\theta_k}-\frac{x_i+0.5}{\theta_i}+\frac{x_k+0.5}{\theta_k}=0$$

resulting in:

$$\theta_i=\theta_k\left(\frac{x_i+\alpha_i-0.5}{x_k+\alpha_k-0.5}\right)$$

To remove θ_k from the estimator, apply $\sum_{i=1}^{k}\theta_i = 1$, which gives:

$$\sum_{i=1}^{k}\theta_k\left(\frac{x_i+\alpha_i-0.5}{x_k+\alpha_k-0.5}\right)=1 \implies \left(\frac{\theta_k}{x_k+\alpha_k-0.5}\right)=\left(\frac{1}{N+\sum_k^{i=1}\alpha_i-\frac{k}{2}}\right)$$

Finally, the MML estimate for θ_i under a *Dirichlet prior* can be written in the following closed form:

$$\theta_i = \left(\frac{x_i + \alpha_i - 0.5}{N + \sum_{i=1}^{k} \alpha_i - \frac{k}{2}} \right) \tag{3.25}$$

In summary, the concepts described in this chapter provide the necessary mathematical foundation for the sequence alignment framework proposed in this thesis. The ensuing chapters present the research core to this thesis. They discuss, amongst others, the formulation and development of MML based statistical models and framework for protein sequence alignment, and an estimation of a full set of statistical models for amino acid substitutions and gaps supporting this framework.

Chapter 4

Modelling Protein Alignments

"I have no data yet. It is a capital mistake to theorise before one has data. Insensibly one begins to twist facts to suit theories, instead of theories to suit facts."

– Arthur Conan Doyle
(in The Adventures of Sherlock Holmes: A Scandal in Bohemia)

The main research contributions of this thesis begin with this chapter. This chapter establishes the basic components of the statistical framework for pairwise sequence alignment based on the Minimum Message Length (MML) criterion. It develops three models in this framework: *null model*, *optimal alignment model* and *marginal probability model*. In this context, each of these models are regarded as a distinct, lossless compression scheme to encode a pair of protein sequences in a single message. The fundamental idea is to select the best model which attains the minimum message length overall. The message length reflects the corresponding Shannon information content (the theoretically optimal lower bound).

This chapter is based primarily on the following publication:
Sumanaweera, D., Allison, L. and Konagurthu, A., The bits between proteins, in 2018 Data Compression Conference (pp. 177-186), IEEE.
DOI: 10.1109/DCC.2018.00026
This chapter also includes parts of the material published in:
Sumanaweera, D., Allison, L. and Konagurthu, A.S., 2019. Statistical compression of protein sequences and inference of marginal probability landscapes over competing alignments using finite state models and Dirichlet priors. Bioinformatics, 35(14), pp.i360-i369.
DOI: 10.1093/bioinformatics/btz368

4.1 MML Framework for Optimal Sequence Alignment

Here we discuss the mathematical details of constructing an MML framework for the problem of pairwise protein sequence alignment.

Overview of the framework for alignment selection: Earlier we saw the general two-part message length formulation of MML to estimate the joint message length of stating an

hypothesis H and data D (Equation 3.10). Here, the data are the pair of amino acid sequences $\langle \mathbf{S}, \mathbf{T} \rangle$. The hypothesis is the statement of any alignment relationship $\mathcal{A}$ between them. This results in the following two-part message length formulation:

$$I(\mathcal{A},\langle \mathbf{S},\mathbf{T} \rangle) = \underbrace{I(\mathcal{A})}_{\text{First part}} + \underbrace{I(\langle \mathbf{S},\mathbf{T} \rangle \mid \mathcal{A})}_{\text{Second part}} \quad \text{bits} \tag{4.1}$$

The first part accounts for communicating losslessly the alignment hypothesis $\mathcal{A}$ describing the sequence relationship, while the second part refers to communicating all amino acid symbols of $\mathbf{S}$ and $\mathbf{T}$ based on the stated relationship in the first part. As introduced in §2.3, a pairwise alignment relationship defines a three-state string over `match` (`m`), `insert` (`i`), and `delete` (`d`) states, generated by a finite state machine defined over those states (see Figure 4.1).

The message length term $I(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$ is associated with the joint probability of $\mathcal{A}$ and $\langle \mathbf{S}, \mathbf{T} \rangle$, and allows the comparison of competing alignment hypotheses. Formally, the difference of alignment model message lengths between any two alignments, $\mathcal{A}_1$ and $\mathcal{A}_2$, gives the log-odds of their posterior probabilities:

$$\begin{aligned} I(\mathcal{A}_1, \langle \mathbf{S}, \mathbf{T} \rangle) - I(\mathcal{A}_2, \langle \mathbf{S}, \mathbf{T} \rangle) &= \log_2 \left(\frac{\Pr(\mathcal{A}_2) \Pr(\langle \mathbf{S}, \mathbf{T} \rangle \mid \mathcal{A}_2)}{\Pr(\mathcal{A}_1) \Pr(\langle \mathbf{S}, \mathbf{T} \rangle \mid \mathcal{A}_1)} \right) \\ &= \log_2 \left(\frac{\Pr(\langle \mathbf{S}, \mathbf{T} \rangle) \Pr(\mathcal{A}_2 \mid \langle \mathbf{S}, \mathbf{T} \rangle)}{\Pr(\langle \mathbf{S}, \mathbf{T} \rangle) \Pr(\mathcal{A}_1 \mid \langle \mathbf{S}, \mathbf{T} \rangle)} \right) = \underbrace{\log_2 \left(\frac{\Pr(\mathcal{A}_2 \mid \langle \mathbf{S}, \mathbf{T} \rangle)}{\Pr(\mathcal{A}_1 \mid \langle \mathbf{S}, \mathbf{T} \rangle)} \right)}_{\text{log-odds posterior ratio}} \end{aligned}$$

It follows from above that the best alignment hypothesis under this information-theoretic framework is the one with the shortest value for $I(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$:

$$\mathcal{A}^* = \underset{\forall \mathcal{A} \in \mathbf{A}}{\operatorname{argmin}} \{ I(\mathcal{A},\langle \mathbf{S},\mathbf{T} \rangle) \} \tag{4.2}$$

Also, as discussed in the previous chapter (§3.3), MML provides a natural test of statistical significance for any hypothesis using a *null model*. Here the null model length $I_{\text{NULL}}(\langle \mathbf{S}, \mathbf{T} \rangle)$ gives the length of a lossless encoding that states each sequence $\mathbf{S}$ and $\mathbf{T}$ independently:

$$I_{\text{NULL}}(\langle \mathbf{S}, \mathbf{T} \rangle) = I_{\text{NULL}}(\mathbf{S}) + I_{\text{NULL}}(\mathbf{T}) \quad \text{bits} \tag{4.3}$$

The full details of estimating a null model for any given sequence of amino acids are described in §4.4. Defining the null model yields the following test of statistical significance:

$$\Delta I_{\text{Optimal}} = I_{\text{NULL}}(\langle \mathbf{S},\mathbf{T} \rangle) - I(\mathcal{A}^*,\langle \mathbf{S},\mathbf{T} \rangle) \quad \text{bits} \tag{4.4}$$

If $\Delta I_{\text{Optimal}} > 0$, we accept the *optimal alignment model* as significant (and $2^{\Delta I_{\text{Optimal}}}$ times more probable than the null (unrelated) statement). Otherwise it is rejected.

4.1.1 Details of Estimating $I(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$

This section describes the estimation of the individual terms in Equation 4.1.

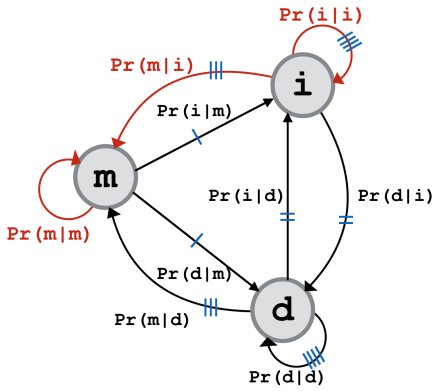


Figure 4.1: A symmetric three-state machine models alignment three-state strings. State transitions associated with three free parameters are highlighted in red. Blue lines on edges show equivalences between various state transition probabilities when insert and delete states are treated symmetrically.

Computation of $I(\mathcal{A})$

This term deals with the estimation of the lossless encoding length of any alignment hypothesis $\mathcal{A}$. Throughout this thesis, $\mathcal{A}$ denotes a three-state alignment string.

The lossless encoding of any alignment $\mathcal{A}$ is composed of the following pieces of information: (1) the length of the 3-state string, (2) the (free) parameters of the 3-state machine over which the alignment is being encoded, and (3) the encoding of each state in $\mathcal{A}$ given those parameters. The total encoding length of these pieces of information is denoted as:

$$I(\mathcal{A}) = I(|\mathcal{A}|) + I(\vec{\Theta}) + I(\mathcal{A}|\vec{\Theta}, |\mathcal{A}|) \quad \text{bits} \tag{4.5}$$

The length of the three-state alignment string $|\mathcal{A}|$ is sent using the relevant Wallace tree integer code (Wallace and Patrick, 1993) described in §3.2.5.

As shown in Figure 4.1, this three-state machine has in total nine state transition probability parameters (i.e. three transitions for each state): $\Pr(\mathtt{m}|\mathtt{m})$, $\Pr(\mathtt{i}|\mathtt{m})$, $\Pr(\mathtt{d}|\mathtt{m})$, $\Pr(\mathtt{m}|\mathtt{i})$, $\Pr(\mathtt{i}|\mathtt{i})$, $\Pr(\mathtt{d}|\mathtt{i})$, $\Pr(\mathtt{m}|\mathtt{d})$, $\Pr(\mathtt{i}|\mathtt{d})$, and $\Pr(\mathtt{d}|\mathtt{d})$.[1,2] Note that the outgoing transition probabilities of each state add up to 1. Further, this 3-state alignment machine is made to be symmetric in terms of `i` and `d` states. This symmetry ensures:

- $\Pr(\mathtt{i}|\mathtt{m}) = \Pr(\mathtt{d}|\mathtt{m}) \equiv \dfrac{1 - \Pr(\mathtt{m}|\mathtt{m})}{2}$;
- $\Pr(\mathtt{d}|\mathtt{i}) = \Pr(\mathtt{i}|\mathtt{d}) \equiv 1 - \Pr(\mathtt{i}|\mathtt{i}) + \Pr(\mathtt{m}|\mathtt{i})$;
- $\Pr(\mathtt{m}|\mathtt{i}) = \Pr(\mathtt{m}|\mathtt{d}))$; and
- $\Pr(\mathtt{i}|\mathtt{i}) = \Pr(\mathtt{d}|\mathtt{d})$.

Altogether, these equivalences reduce the number of free parameters to just three (notionally: $\vec{\Theta} = \{\Pr(\mathtt{m}|\mathtt{m}), \Pr(\mathtt{i}|\mathtt{i}), \Pr(\mathtt{m}|\mathtt{i})\}$), from which all the remaining six dependent parameters can be inferred. Estimation of these three free parameters is done as follows. For the `match` state, it involves a point-estimation in a 1-simplex: $[\Pr(\mathtt{m}|\mathtt{m}), 1 - \Pr(\mathtt{m}|\mathtt{m})]$. For the `insert` state, it

[1] The first character of any string from this three-state machine is encoded with a probability of $\frac{1}{3}$.

[2] In many alignment schemes, a transition between `i` and `d` states is forbidden, implying a transition probability of zero. This thesis deals with the inference of a complete finite state machine, which includes the transition probabilities $\Pr(\mathtt{i}|\mathtt{d})$ and $\Pr(\mathtt{d}|\mathtt{i})$ without constraining them artificially.

involves a point-estimation in a 2-simplex: $[\Pr(\mathtt{i}|\mathtt{i}), \Pr(\mathtt{m}|\mathtt{i}), 1 - \Pr(\mathtt{i}|\mathtt{i}) - \Pr(\mathtt{m}|\mathtt{i})]$. There are two approaches to estimate the free-parameter vector $\vec{\Theta}$ of the alignment 3-state machine, which is then used to encode any alignment string and estimate $I(\mathcal{A})$:

Approach 1: Use the MML87 method of parameter estimation for $\vec{\Theta}$ based on the evidence of the three states in $\mathcal{A}$. Note, this approach can use either uniform priors (as per Equation 3.24) or any given Dirichlet priors (as per Equation 3.25).

Approach 2: Use the MML87 method of parameter estimation for inferring divergence 'time' dependent Dirichlet distributions from any given alignment benchmark. Subsequently, apply the expected values under the inferred Dirichlet models to parameterise $\vec{\Theta}$.

This Chapter deals with **Approach 1**, applied using a uniform prior. **Approach 2** is explored in detail in Chapter 5, where the MML based Dirichlet model estimation is discussed.

Finally, if the alignment $\mathcal{A} = \{\langle state\rangle_1 \langle state\rangle_2 \langle state\rangle_3 \cdots \langle state\rangle_{|\mathcal{A}|}\}$, where any $\langle state\rangle_i \in \{\mathtt{m}, \mathtt{i}, \mathtt{d}\}$, then:

$$I(\mathcal{A}|\vec{\Theta}) = -\log_2\left(\frac{1}{3}\right) + \sum_{i=2}^{|\mathcal{A}|} -\log_2(\Pr(\langle state\rangle_i \,|\, \langle state\rangle_{i-1})) \quad \text{bits}$$

Computation of $I(\langle \mathbf{S}, \mathbf{T}\rangle \,|\mathcal{A})$:

This involves the message-length term required to explain all amino acids in the sequence pair $\langle \mathbf{S}, \mathbf{T}\rangle$, using the knowledge of their alignment relationship $\mathcal{A}$. Solely based on the information of $\mathcal{A}$, the length of $\mathbf{S}$ (the number of '$\mathtt{m}$'s plus the number of '$\mathtt{d}$'s in $\mathcal{A}$) and the length of $\mathbf{T}$ (the number of '$\mathtt{m}$'s plus the number of '$\mathtt{i}$'s in $\mathcal{A}$) are known. The details of the amino acid symbols that make up these sequences should be accounted for. To achieve this, we estimate the length of encoding the amino acid symbols associated with each alignment state.

Specifically, the $\mathtt{m}$ state implies an alignment between pairs of amino acids of the form $\mathbf{S}_i$ and $\mathbf{T}_j$. The $\mathtt{d}$ state implies that $\mathbf{S}_i$ remains unaligned, whereas the $\mathtt{i}$ state implies that $\mathbf{T}_j$ remains unaligned.

Encoding amino acids under insert or delete state

The statement of amino acids in $\mathtt{i}$ or $\mathtt{d}$ state is carried out using the amino acid probabilities coming under the *null model*, taking $I_{\text{NULL}}(\mathbf{S}_i)$ and $I_{\text{NULL}}(\mathbf{T}_j)$ bits, respectively. (The details of inferring the *null model* are given in §4.4).

Encoding amino acid pairs under match states

For the $\mathtt{m}$ state, each pair of amino acids $\langle \mathbf{S}_i, \mathbf{T}_j\rangle$ are encoded using a probabilistic model of amino acid substitution.

This chapter relies on an existing probabilistic model of amino acid substitution, which is the PAM Markov model of substitution by Dayhoff et al. (1978). (Later in Chapter 6, new Markov models are inferred under the MML framework for the same purpose). Dayhoff and colleagues modelled the transition probabilities of each amino acid by relying on families of closely-related proteins. Their approach derived an evolutionary time unit called Point Accepted Mutation with a matrix called PAM-1. This matrix (denoted by $\mathbf{M}^1$) is a conditional probability matrix

over all 20 amino acid symbols, where any $\mathbf{M}^1(i,j)$ gives $\Pr(aa_i|aa_j)$: the transition probability of the amino acid aa_j mutating into the amino acid aa_i, in one PAM-step. This is same as the probability that amino acid aa_j mutates into an amino acid aa_i between sequences that have diverged by one PAM unit of time. (Note, $\mathbf{M}^1(i,j) \equiv \Pr(aa_i|aa_j) \neq \Pr(aa_j|aa_i) \equiv \mathbf{M}^1(j,i)$.)

Generalising, a PAM-t matrix ($\mathbf{M}^t \quad \forall t > 0$) gives the probability of any amino acid aa_j mutating into any amino acid aa_i in t PAM units of time. PAM-t can be derived from PAM-1 ($\mathbf{M}^1$) via *matrix exponentiation*: $(\mathbf{M}^1)^t = \mathbf{M}^t$. Further details on the mathematical properties of this model are discussed comprehensively in Chapter 6. Note: The framework facilitates any Markov model of amino acid substitution (such as PAM) to be applied as $\mathbf{M}^t$.

Given an alignment $\mathcal{A}$ of a sequence pair $\langle \mathbf{S}, \mathbf{T} \rangle$, let $\mathcal{A}_{\mathtt{m}}$ denote the subsequence of $\mathcal{A}$ that defines all its matches. Denote the ordered set of matches between amino acid pairs contained in the subsequence $\mathcal{A}_{\mathtt{m}}$ to be of the form: $\langle \mathbf{S}, \mathbf{T} \rangle_{\mathtt{m}} = (\langle \mathbf{S}_{i_1}, \mathbf{T}_{j_1} \rangle, \langle \mathbf{S}_{i_2}, \mathbf{T}_{j_2} \rangle, \ldots)$. There are two approaches to encode these amino acid pairs using a given PAM-t ($\mathbf{M}^t$) substitution matrix:

(i) Amino acid mutation-generation (asymmetric) machine:

Here, each matched pair of symbols $\langle \mathbf{S}_{i_k}, \mathbf{T}_{j_k} \rangle \in \langle \mathbf{S}, \mathbf{T} \rangle_{\mathtt{m}}$ is stated as follows. First, $\mathbf{T}_{j_k}$ is stated using the *null model* probability of $\Pr(\mathbf{T}_{j_k})$, taking: $I_{\text{NULL}}(\mathbf{T}_{j_k}) = -\log_2(\Pr(\mathbf{T}_{j_k}))$ bits. Then, $\mathbf{S}_{i_k}$ is stated using $\mathbf{T}_{j_k}$ and PAM-t, taking $I(\mathbf{S}_{i_k}|\mathbf{T}_{j_k}, \mathbf{M}^t) = -\log_2(\Pr(\mathbf{S}_{i_k}|\mathbf{T}_{j_k}, \mathbf{M}^t))$ $= -\log_2(\mathbf{M}^t(\mathbf{S}_{i_k}, \mathbf{T}_{j_k}))$ bits. Thus, over the set of all matches defined by $\mathcal{A}_{\mathtt{m}}$, the message length term $I(\langle \mathbf{S}, \mathbf{T} \rangle | \mathcal{A}_{\mathtt{m}})$ under this machine is:

$$\begin{aligned} I(\langle \mathbf{S}, \mathbf{T} \rangle | \mathcal{A}_{\mathtt{m}}) &= \sum_{k=1}^{|\mathcal{A}_{\mathtt{m}}|} -\log_2(\Pr(\mathbf{S}_{i_k}, \mathbf{T}_{j_k}|\mathbf{M}^t)) \\ &= \sum_{k=1}^{|\mathcal{A}_{\mathtt{m}}|} \left(I_{\text{NULL}}(\mathbf{T}_{j_k}) + I(\mathbf{S}_{i_k}|\mathbf{T}_{j_k}, \mathbf{M}^t) \right) \quad \text{bits.} \end{aligned} \tag{4.6}$$

This approach is asymmetric in general. That means, swapping the order of sequences from $\langle \mathbf{S}, \mathbf{T} \rangle$ to $\langle \mathbf{T}, \mathbf{S} \rangle$ changes the message length, unless the stationary distribution of the Markov substitution matrix is used as the *null* probability distribution of amino acids. (Such symmetry is possible due to the properties of Markov Models discussed in §6.2).

(ii) Amino acid pair-generation (symmetric) machine:

The above mentioned asymmetry encountered in the mutation-generation machine can be overcome by constructing the average of the (asymmetric) joint probabilities of $\Pr(\mathbf{S}_{i_k}, \mathbf{T}_{j_k}|\mathbf{M}^t)$ and $\Pr(\mathbf{T}_{j_k}, \mathbf{S}_{i_k}|\mathbf{M}^t)$ as:

$$\overset{\text{avg}}{\Pr}(\mathbf{S}_{i_k}, \mathbf{T}_{j_k}|\mathbf{M}^t) = \frac{(\Pr(\mathbf{S}_{i_k}) \Pr(\mathbf{T}_{j_k}|\mathbf{S}_{i_k}, \mathbf{M}^t) + \Pr(\mathbf{T}_{j_k}) \Pr(\mathbf{S}_{i_k}|\mathbf{T}_{j_k}, \mathbf{M}^t))}{2}$$

which remains invariant to the order of evaluation of the two sequences. Thus:

$$I(\langle \mathbf{S}, \mathbf{T} \rangle | \mathcal{A}_{\mathtt{m}}) = \sum_{k=1}^{|\mathcal{A}_{\mathtt{m}}|} -\log_2(\overset{\text{avg}}{\Pr}(\mathbf{S}_{i_k}, \mathbf{T}_{j_k}|\mathbf{M}^t)) = \sum_{k=1}^{|\mathcal{A}_{\mathtt{m}}|} I(\mathbf{S}_{i_k}, \mathbf{T}_{j_k}|\mathbf{M}^t) \quad \text{bits.} \tag{4.7}$$

Estimating the optimal evolutionary time for a given alignment:

Given an alignment $\mathcal{A}$ over a sequence pair $\langle \mathbf{S}, \mathbf{T} \rangle$, we want to select the best evolutionary time t under the PAM model (i.e. $\mathbf{M}^t$ = PAM-t) such that, the term $I(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$ computed using either Equation 4.6 or Equation 4.7 is minimised. Equation 4.9 gives the complete breakdown of the term. Here, we restrict t to be a positive integer, and attempt to minimise $I(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$ using a variant of the iterative bisection search over the domain $L = 1 \leq t \leq 1000 = U$. This procedure is further elaborated in §4.1.2 below, under the 'Expectation-Maximisation like approach'.

Finally, the statement length of the second part of the message becomes:

$$I(\langle \mathbf{S}, \mathbf{T} \rangle | \mathcal{A}) = \sum_{\forall \langle \mathbf{S}_i, \mathbf{T}_j \rangle \in \mathcal{A}_{\mathtt{m}}} I(\mathbf{S}_i, \mathbf{T}_j | \mathbf{M}^t) + \sum_{\forall \mathbf{S}_i \in \mathcal{A}_{\mathtt{d}}} I(\mathbf{S}_i) + \sum_{\forall \mathbf{T}_j \in \mathcal{A}_{\mathtt{i}}} I(\mathbf{T}_j) \quad \text{bits} \tag{4.8}$$

4.1.2 Search for the Optimal Alignment

The aforementioned encoding scheme of the *alignment model* is applicable to any given alignment $\mathcal{A}$. However, we are interested in finding the optimal alignment $\mathcal{A}^*$ that minimises the two part message length $I(\mathcal{A}^*, \langle \mathbf{S}, \mathbf{T} \rangle)$, expressed via Equation 4.2. This involves a simultaneous finding of the optimal values for its associated parameters $\{\vec{\Theta}_{\text{opt}}, t_{\text{opt}}\}$. The goal is the optimal inference of $\{\mathcal{A}, \vec{\Theta}, t\}$, through an Expectation-Maximisation (EM) like iterative procedure. At each iteration step of EM, the best alignment is found for some *fixed* set of $\{\vec{\Theta}, t\}$ using a Dynamic Programming (DP) approach. Then the best parameters are inferred for that particular alignment. These will be the fixed parameters for the next iteration. The search procedure continues until convergence.

Dynamic Programming Algorithm

Given a fixed parameter set $\{\vec{\Theta}, t\}$, the objective is to find the optimal alignment that minimises

$$I(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle) = I(|\mathcal{A}|) + I(t) + I(\vec{\Theta}) + I(\mathcal{A}|\vec{\Theta}) + I(\langle \mathbf{S}, \mathbf{T} \rangle | \mathcal{A}) \tag{4.9}$$

using Dynamic Programming (DP). To permit strict-additivity that is required to implement DP, $I(|\mathcal{A}|)$ is considered as a constant for all practical purposes. This is justified because:

(i) closely-competing alignments around the optimal almost always have the same alignment lengths, and hence yield the same value for $I(|\mathcal{A}|)$; and

(ii) furthermore, $I(\mathcal{A}|\vec{\Theta}) + I(\langle \mathbf{S}, \mathbf{T} \rangle | \mathcal{A}) \gg I(|\mathcal{A}|)$

The framework adopts a DP strategy similar to the one proposed by Gotoh (1982) as described in §2.4.2. Three history matrices, $\text{Hist}_{\mathtt{m}}$, $\text{Hist}_{\mathtt{i}}$ and $\text{Hist}_{\mathtt{d}}$ of size $|\mathbf{S}| + 1 \times |\mathbf{T}| + 1$ are built for each state $\in \{\mathtt{m},\mathtt{i},\mathtt{d}\}$. Any $\text{Hist}_{\mathtt{m}}(i, j)$ accounts for the alignment of $\mathbf{S}_{1:i}$ and $\mathbf{T}_{1:j}$ prefixes that ends in an $\mathtt{m}$ state. Similarly, $\text{Hist}_{\mathtt{i}}(i, j)$ and $\text{Hist}_{\mathtt{d}}(i, j)$ denote the message lengths of communicating alignments of those prefixes that end in state $\mathtt{i}$ and state $\mathtt{d}$, respectively. The recurrence relations are defined below.

$$\text{Hist}_{\mathtt{m}}(i,j) = \mathtt{min}\begin{cases}\text{Hist}_{\mathtt{m}}(i-1,j-1) + I_{\vec{\Theta}}(\mathtt{m}|\mathtt{m}) + I(\mathbf{S}_i,\mathbf{T}_j|\mathbf{M}^t)\\ \text{Hist}_{\mathtt{i}}(i-1,j-1) + I_{\vec{\Theta}}(\mathtt{m}|\mathtt{i}) + I(\mathbf{S}_i,\mathbf{T}_j|\mathbf{M}^t)\\ \text{Hist}_{\mathtt{d}}(i-1,j-1) + I_{\vec{\Theta}}(\mathtt{m}|\mathtt{d}) + I(\mathbf{S}_i,\mathbf{T}_j|\mathbf{M}^t)\end{cases}$$

$$\text{Hist}_{\mathtt{i}}(i,j) = \mathtt{min}\begin{cases}\text{Hist}_{\mathtt{m}}(i-1,j) + I_{\vec{\Theta}}(\mathtt{i}|\mathtt{m}) + I(\mathbf{S}_i)\\ \text{Hist}_{\mathtt{i}}(i-1,j) + I_{\vec{\Theta}}(\mathtt{i}|\mathtt{i}) + I(\mathbf{S}_i)\\ \text{Hist}_{\mathtt{d}}(i-1,j) + I_{\vec{\Theta}}(\mathtt{i}|\mathtt{d}) + I(\mathbf{S}_i)\end{cases}$$

$$\text{Hist}_{\mathtt{d}}(i,j) = \mathtt{min}\begin{cases}\text{Hist}_{\mathtt{m}}(i,j-1) + I_{\vec{\Theta}}(\mathtt{d}|\mathtt{m}) + I(\mathbf{T}_j)\\ \text{Hist}_{\mathtt{i}}(i,j-1) + I_{\vec{\Theta}}(\mathtt{d}|\mathtt{i}) + I(\mathbf{T}_j)\\ \text{Hist}_{\mathtt{d}}(i,j-1) + I_{\vec{\Theta}}(\mathtt{d}|\mathtt{d}) + I(\mathbf{T}_j)\end{cases}$$

Finally, the optimal message length is computed as:

$$I(\mathcal{A}^*,\langle\mathbf{S},\mathbf{T}\rangle) = I(|\mathcal{A}^*|) + I(t_{\text{opt}}) + I(\vec{\Theta}_{\text{opt}}) + \underbrace{\mathtt{min}\{\text{Hist}_{\mathtt{m}}(|\mathbf{S}|,|\mathbf{T}|), \text{Hist}_{\mathtt{i}}(|\mathbf{S}|,|\mathbf{T}|), \text{Hist}_{\mathtt{d}}(|\mathbf{S}|,|\mathbf{T}|)\}}_{\text{bits}}$$

The optimal alignment $\mathcal{A}^*$ can be constructed by tracing back across the history matrices, starting from the cell $(|\mathbf{S}|,|\mathbf{T}|)$ of the Hist matrix that gives the minimum message length. In case of ties, the order `m`,`i` and `d` is followed.

Note: In this thesis, an insertion denotes alignment columns where gaps occur in $\mathbf{T}$ as opposed to amino acids (inserted) in $\mathbf{S}$. A deletion is the other way around.

Expectation-Maximisation like approach for Optimal $\{\vec{\Theta}, t\}$

The EM-like process starts with an initial set of parameters $\{\vec{\Theta}, t\}$. The *Expectation step (E-step)* involves running the DP algorithm described in §4.1.2, leading to an initial alignment $\mathcal{A}$. Next, the *maximisation step (M-step)* performs a separate maximisation for $\vec{\Theta}$ and t independently, by keeping the currently obtained $\mathcal{A}$ fixed. Upon the M-step, the parameters are updated, and the EM process is repeated until the alignment does not change anymore (i.e. convergence). Parameters are updated as follow.

(1) Updating the optimal time t: Given the mapping between evolutionary time and substitution probability matrices defined in the *codebook*, the search for the optimal evolutionary time t goes as follows. The framework in this chapter searches for the best $t \in [1, t_{\max}]$ that minimises the message length associated with the likelihood of all matches $\sum_{\forall\langle\mathbf{S}_i,\mathbf{T}_j\rangle\in\mathcal{A}_{\mathtt{m}}} I(\mathbf{S}_i,\mathbf{T}_j\ |\mathbf{M}^t)$, under the currently fixed $\mathcal{A}$ using a substitution matrix $\mathbf{M}^t$ (as briefly introduced in §4.1.1). An exhaustive grid search is inefficient for this search, as it takes $\text{O}(t_{\max}|\mathcal{A}|)$ time to find the minimum. A logarithmic time-complexity search algorithm such as bisection search or a gradient descent method can be used for the purpose. This thesis initially employed a variant of the bisection search (i.e. a quaternary search) over the domain $L = 1 \leq t \leq U = t_{\max}$, where $t_{\max=1000}$.[3] For the fixed set of matches present in an alignment, in each iteration, the algorithm truncates the search domain $[L,U]$ to $[L + \frac{U-L}{4}, U]$ or $[L, U - \frac{U-L}{4}]$ by removing the first or the last quarter, based on the value evaluated for the associated message length. If the message length at the lower bound is higher than the upper bound, the lower bound is updated.

[3] Choosing $t_{\max}$ was based on the observed time taken for a Markov substitution model to reach the stationary distribution across a selected set of Markov matrices. This is reasoned in Chapter 6.

Otherwise, the upper bound is updated. The iterations stop when the search converges, and the value of t is updated with the inferred time measure.

(2) Updating $\vec{\Theta}$: This is dependent on which approach we use for parameterising $\vec{\Theta}$ (as defined in 4.1.1). Under **Approach 1**, the three-state machine parameters are re-estimated using the MML87 method over the observed alignment string. Under **Approach 2**, the updated t automatically gives the new values for $\vec{\Theta}$ (drawn from a mapping between evolutionary time t and its suitable $\vec{\Theta}$ estimates under the *codebook*). In the results presented here, we use **Approach 1**. In the next chapter, with further extensions, the use of **Approach 2** is demonstrated.

Computational Complexity

The overall search for an optimal alignment under the methods described above takes $O(|\mathbf{S}||\mathbf{T}|)$ effort to compute. The total effort is a function of:

- the number of EM iterations: n_{EM}
- the time taken to fill the three history matrices: $O(|\mathbf{S}||\mathbf{T}|)$
- the time taken to find the optimal time parameter t in each EM iteration: $O(|\mathcal{A}|) = O(|\mathbf{S}| + |\mathbf{T}|)$
- the time taken to find the three-state machine parameters in each EM iteration, optimal for the current alignment: $O(|\mathcal{A}|) = O(|\mathbf{S}| + |\mathbf{T}|)$.

In practice, n_{EM} is a small integer (in most cases ≤ 5). Note: In any iteration, the length of an alignment ($|\mathcal{A}|$) is bounded by $|\mathbf{S}| + |\mathbf{T}|$. Separately, the space requirement of maintaining the three history matrices is $O(|\mathbf{S}||\mathbf{T}|)$.

4.1.3 Performance of the Initial Alignment Model

The above detailed MML framework for optimal sequence alignment enabled an initial evaluation to test the effectiveness of its unsupervised protein alignment under both *symmetric* and *asymmetric* machines (described in §4.1.1). A benchmark of 630 pairwise alignments were downloaded from the manually-curated HOMologous STRucture Alignment Database (HOMSTRAD) database (Mizuguchi et al., 1998) for this purpose. They were realigned using the MML framework, applying PAM (Dayhoff et al., 1978) Markov model as the amino acid substitution model $\mathbf{M}$. Let us denote the PAM series by $\texttt{PAM} = \{\mathbf{M}^1, \mathbf{M}^2, ..., \mathbf{M}^{t_{\text{Max}}}\}$, where a conditional probability matrix is defined for each discrete evolutionary time $t \in [1, t_{\text{Max}}]$, taking $t_{\text{Max}} = 1000$. (Note: All computational aspects, justifications and further improvements related to the substitution matrix series are detailed in Chapter 6).

The 630 pairs were also realigned using two popular alignment programs, ClustalW2 (Larkin, 2007) and MUSCLE (Edgar, 2004), in the pairwise alignment mode under their default parameter settings. Further, the general Gotoh alignment algorithm (Gotoh, 1982) with affine gap penalty function $\Gamma(l) = \mathbf{g}_o + (l-1)\mathbf{g}_e$ was also included in this study. Since it works with a fixed scoring matrix and gap penalties, the evaluation was done under the $\langle \mathbf{g}_o = 10, \mathbf{g}_e = 1 \rangle$ setting with $\{\mathbf{M}^{10}, \mathbf{M}^{100}, \mathbf{M}^{250}\}$ matrices from the PAM series, as well three other non-Markov substitution matrices, {BLOSUM-30, BLOSUM-62, BLOSUM-90} from another widely-used matrix series BLOSUM (Henikoff and Henikoff, 1992).

Table 4.1: The 1st (Q1), 2nd (Q2=median) and third (Q3) quartile statistics of compression gain observed across the alignments generated by various methods for 630 pairs of proteins in the HOMSTRAD benchmark

	Compression (in bits)					
	Symmetric			**Asymmetric**		
	Q1	**Q2**	**Q3**	**Q1**	**Q2**	**Q3**
ClustalW2	4.8	72.5	231.5	2.2	69.5	229.3
MUSCLE	5.8	77.5	232.2	4.9	75.1	230.3
Gotoh + BLOSUM-30	-10.5	66.6	221.6	-11.9	62.9	219.6
Gotoh + BLOSUM-62	1.8	74.7	232.2	-0.8	73.4	225.2
Gotoh + BLOSUM-90	-7.5	69.9	225.0	-9.8	66.5	222.9
Gotoh + PAM-10	-35.5	15.1	177.1	-35.3	14.5	175.9
Gotoh + PAM-100	-3.5	71.3	227.4	-5.5	70.3	224.2
Gotoh + PAM-250	3.7	75.5	228.1	2.2	72.9	227.6
$\text{MML}_{\text{Optimal}}$	**12.6**	**82.7**	**233.6**	**11.5**	**82.4**	**233.1**

Alignments generated by different programs were compared in terms of their compression gain under the *optimal alignment model* (Δ_{Optimal}) defined in the Equation 4.4. To allow this, *optimal alignment model* parameters $\{\vec{\Theta}, t\}$ were inferred for every alignment across other programs using the *M-step* described in §4.1.2. Separately, the *null model* probability distribution of amino acids was estimated using the MML87 method over a set of complete protein sequences from the UniProt (UniProt Consortium and others, 2017) database (See §4.4 for details on *null model* inference). These enable the computation of: $I_{\text{NULL}}(\langle \mathbf{S}, \mathbf{T} \rangle) - I(\mathcal{A}^*, \langle \mathbf{S}, \mathbf{T} \rangle)$.

Table 4.1 presents the quartile statistics of compression, under both symmetric and asymmetric machines of amino acid pairs. This table shows that the alignments inferred using the MML based *optimal alignment model* yields the most compression, consistently across all quartile marks.

Note: This section presented only the preliminary results of the premature MML protein optimal alignment model introduced in this chapter. Later chapters 5 and 6 evolve the framework further with better models of alignment and amino acid evolution, and present more comprehensive results on their evaluation over several benchmarks compared to several popular alignment programs and existing substitution models.

4.2 MML Framework to *Marginalise* over All Possible Alignments

As an alternative to the optimal alignment framework explored in the previous section, this section explores the use of MML framework to compute the marginal probability (described by Equation 3.3) in Shannon information terms, over all possible alignments between two protein sequences.

Marginal (or total) probability estimation applied to sequence alignment provides a powerful technique to highlight the relationship between sequences, by 'marginalising out' all the alignments between them (Allison and Wallace, 1994; Eddy, 1998). This is useful when proteins have far diverged beyond a point where the optimal alignment is unable to capture statistically-significant similarities. Rost (1999) comments on the difference between accurate evolutionary

relationship detection and alignment as follows: *"Did this imply that correct detection and correct alignment were not correlated?... Not necessarily, but the fact is that two homologues can be detected although part or even the entire alignment is wrong"*

Here we define a *marginal probability model* under the MML framework which is more general in evaluating the evolutionary relationship between $\mathbf{S}$ and $\mathbf{T}$ without specifying an alignment. It is also used for effective visualisations of competing alignments (See §4.3.2). This enables us to answer the question: *Are two protein sequences evolutionarily related?*. The total law of probability (presented in §3.1 with marginal probability definition given by the Equation 3.3) allows us to integrate over the complete alignment space, in order to compute the *marginal probability model* as follows:

$$I_{\text{Marginal}}(\langle \mathbf{S},\mathbf{T} \rangle) = -\log_2 \left(\sum_{\forall \mathcal{A} \in \mathbf{A}} \Pr(\mathcal{A}, \langle \mathbf{S},\mathbf{T} \rangle) \right) \quad \text{bits} \tag{4.10}$$

This model can be explored in a similar fashion to the *optimal alignment model*, yielding the following statistical significance test with respect to a *null model*:

$$\Delta I_{\text{Marginal}} = I_{\text{NULL}}(\langle \mathbf{S},\mathbf{T} \rangle) - I_{\text{Marginal}}(\langle \mathbf{S},\mathbf{T} \rangle) \quad \text{bits} \tag{4.11}$$

If $\Delta I_{\text{Marginal}} > 0$, we accept the *marginal probability model* as indicating some *unspecified* relationship. Otherwise, it is rejected, concluding that there is no sequence signal that reveals any evolutionary relationship between the pair.

More importantly, the estimate of $I_{\text{Marginal}}(\langle \mathbf{S},\mathbf{T} \rangle)$ can also be implemented using an effective DP approach similar to the *optimal alignment* DP, via three history matrices: $\text{Tot}_{\mathtt{m}}$, $\text{Tot}_{\mathtt{i}}$ and $\text{Tot}_{\mathtt{d}}$. Each cell (i, j) in these matrices stores the negative logarithm of the marginal probability that the prefixes $\mathbf{S}_{1:i}$ and $\mathbf{T}_{1:j}$ are related, by summing over all alignments ending in $\mathtt{m}$, $\mathtt{i}$, and $\mathtt{d}$ states respectively. This can be efficiently computed using the negative log sum of exponentials ($\mathtt{LSE}$) function under the dynamic programming recurrences given below:

$$\text{Tot}_{\mathtt{m}}(i,j) = -\mathtt{LSE} \begin{cases} \text{Tot}_{\mathtt{m}}(i-1,j-1) + I_{\vec{\Theta}}(\mathtt{m}|\mathtt{m}) + I(\mathbf{S}_i,\mathbf{T}_j|\mathbf{M}^t) \\ \text{Tot}_{\mathtt{i}}(i-1,j-1) + I_{\vec{\Theta}}(\mathtt{m}|\mathtt{i}) + I(\mathbf{S}_i,\mathbf{T}_j|\mathbf{M}^t) \\ \text{Tot}_{\mathtt{d}}(i-1,j-1) + I_{\vec{\Theta}}(\mathtt{m}|\mathtt{d}) + I(\mathbf{S}_i,\mathbf{T}_j|\mathbf{M}^t) \end{cases}$$

$$\text{Tot}_{\mathtt{i}}(i,j) = -\mathtt{LSE} \begin{cases} \text{Tot}_{\mathtt{m}}(i-1,j) + I_{\vec{\Theta}}(\mathtt{i}|\mathtt{m}) + I(\mathbf{S}_i) \\ \text{Tot}_{\mathtt{i}}(i-1,j) + I_{\vec{\Theta}}(\mathtt{i}|\mathtt{i}) + I(\mathbf{S}_i) \\ \text{Tot}_{\mathtt{d}}(i-1,j) + I_{\vec{\Theta}}(\mathtt{i}|\mathtt{d}) + I(\mathbf{S}_i) \end{cases}$$

$$\text{Tot}_{\mathtt{d}}(i,j) = -\mathtt{LSE} \begin{cases} \text{Tot}_{\mathtt{m}}(i,j-1) + I_{\vec{\Theta}}(\mathtt{d}|\mathtt{m}) + I(\mathbf{T}_j) \\ \text{Tot}_{\mathtt{i}}(i,j-1) + I_{\vec{\Theta}}(\mathtt{d}|\mathtt{i}) + I(\mathbf{T}_j) \\ \text{Tot}_{\mathtt{d}}(i,j-1) + I_{\vec{\Theta}}(\mathtt{d}|\mathtt{d}) + I(\mathbf{T}_j) \end{cases}$$

Note: The logarithm of the sum of exponentials ($\mathtt{LSE}$) of a set of arguments a, b, c as:

$$\begin{aligned} \mathtt{LSE}\{a,b,c\} &= \log_2(a+b+c) \\ &= \log_2\left(a \cdot \left(\frac{a}{a} + \frac{b}{a} + \frac{c}{a}\right)\right) \\ &= \log_2(a) + \log_2\left(1 + 2^{\log_2(b) - \log_2(a)} + 2^{\log_2(c) - \log_2(a)}\right) \end{aligned}$$

Ultimately, $I_{\text{Marginal}}(\langle \mathbf{S},\mathbf{T} \rangle)$ over all possible alignments (as per the Equation 4.10) is given by:

$$I(\langle \mathbf{S},\mathbf{T} \rangle) = -\texttt{LSE}\{\text{Tot}_{\texttt{m}}(|\mathbf{S}|,|\mathbf{T}|), \text{Tot}_{\texttt{i}}(|\mathbf{S}|,|\mathbf{T}|), \text{Tot}_{\texttt{d}}(|\mathbf{S}|,|\mathbf{T}|)\} \quad \text{bits}$$

This computation has similarities to the computation done in the *forward-backward* algorithm (Eddy, 1998). Finally, the same EM approach (presented in §4.1.2 for *optimal alignment model*) can be used to optimise for $\{\vec{\Theta}, t\}$.

Search for the Optimal $\{\vec{\Theta}, t\}$

Again, the optimisation routine starts with an initial $\{\vec{\Theta}, t\}$. *E-step* runs the DP algorithm described above. *M-step* maximises the parameters under the fixed *marginal probability model* and updates them for the next iteration. EM process continues until convergence.

However, the *E-step* does not lead to a definitive alignment between the sequences. Thus, the search for the best $t \in [1, t_{\max}]$ is no longer on a fixed alignment, but across all alignments by minimising the total message length $I(\langle \mathbf{S},\mathbf{T} \rangle)$. Further, *Approach 1* for $\vec{\Theta}$ estimation is feasible only if we sample an average alignment from the space and compute optimal $\vec{\Theta}$ using MML multi-state inference by fixing that alignment. For *Approach 2*, the $\vec{\Theta}$ which is mapped to the updated value of evolutionary time t in the *codebook* can be applied. (See §4.1.1 for the introduction of *Approach 1* and *Approach 2* for estimating the free parameter vector $\vec{\Theta}$ of the three-state machine).

Computational Complexity

The *marginal probability model* requires an $\text{O}(|\mathbf{S}||\mathbf{T}|)$ time and space complexity. Mainly, this is because the derivation of the total probability of prefixes of sequences at each cell in the three DP matrices requires only $O(1)$ effort. All other considerations are same as discussed for optimal alignments.

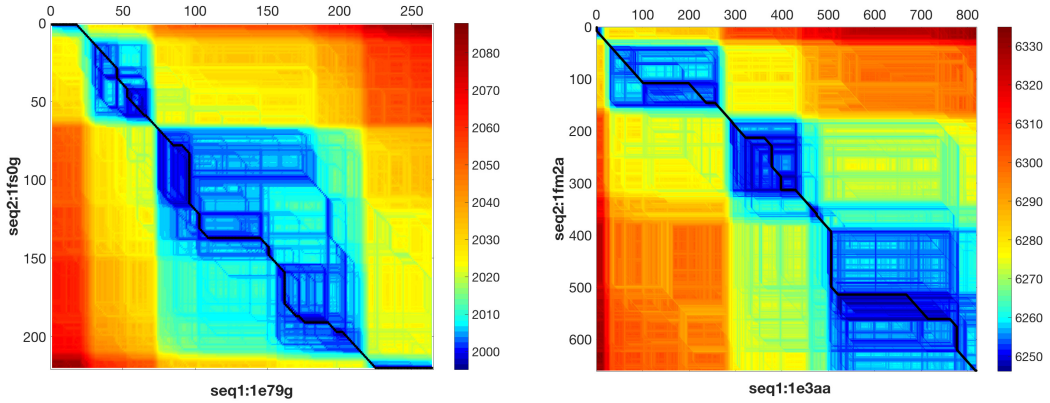
4.3 Alignment Landscapes

Alignment landscapes provide useful visualisations of competing alignments across the the entire space of alignments between pairs of protein sequences. Since all alignments are evaluated probabilistically in this MML framework, such landscape enables users to explore highly probable alternative alignments in addition to the best under this framework. Adopting the same line of thought in the work by Allison and Wallace (1994); Yee and Allison (1993) implemented for DNA sequence alignment, this thesis defines two types of alignment landscapes for protein sequence alignment under the MML framework introduced in §4.1: (1) *optimal alignment landscape* and (2) *marginal probability landscape*.

In a nutshell, an alignment landscape is represented by a single matrix $\mathcal{L}$ of size $(|\mathbf{S}|+1) \times (|\mathbf{T}|+1)$. Each cell $\mathcal{L}(i,j)$ corresponds to a $\langle \mathbf{S}_{1:i}, \mathbf{T}_{1:j} \rangle$ prefix alignment and $\langle \mathbf{S}_{j+1:|\mathbf{S}|}, \mathbf{T}_{i+1:|\mathbf{T}|} \rangle$ suffix alignment. In other words, it is a joint prefix and suffix alignment, passing through a cell (i,j). The landscape matrix $\mathcal{L}$ is constructed as follows:

1. Align $\mathbf{S}$ and $\mathbf{T}$ in the *forward* direction (i.e. align prefixes $\langle \mathbf{S}_{1:|\mathbf{S}|}, \mathbf{T}_{1:|\mathbf{T}|} \rangle$), optimally or marginally through the EM based iterative process described previously under the MML framework

2. Align $\mathbf{S}$ and $\mathbf{T}$ in *backward* direction (i.e. align suffixes $\langle \mathbf{S}_{|\mathbf{S}|:1}, \mathbf{T}_{|\mathbf{T}|:1} \rangle$) in single iteration, under the optimal parameter setting inferred for *forward* alignment



(a) (b)

Figure 4.2: Competing optimal alignment landscapes of (a) γ subunits of ATP Synthases from *Bos Taurus* (PDB ID: `1E79`, chain G) and *Escherichia Coli* (PDB ID: `1FS0`, chain G); (b) cephalosporin acylase from *Brevundimonas diminuta* (PDB ID: `1FM2` chain A) and penicillin acylase from *Escherichia coli* (PDB ID: `1E3A` chain A).

3. Consistently collate the three *forward* DP history matrices and the three *backward* DP history matrices to form $\mathcal{L}$

In *backward* alignment, proper reverse encoding is enforced by reversing the three-state machine. Also this is straightforward only when using a Markov order-0 *null model*. The next two sections present the specifics of the above procedure for the two distinct types of alignment landscapes.

4.3.1 Optimal Alignment Landscape

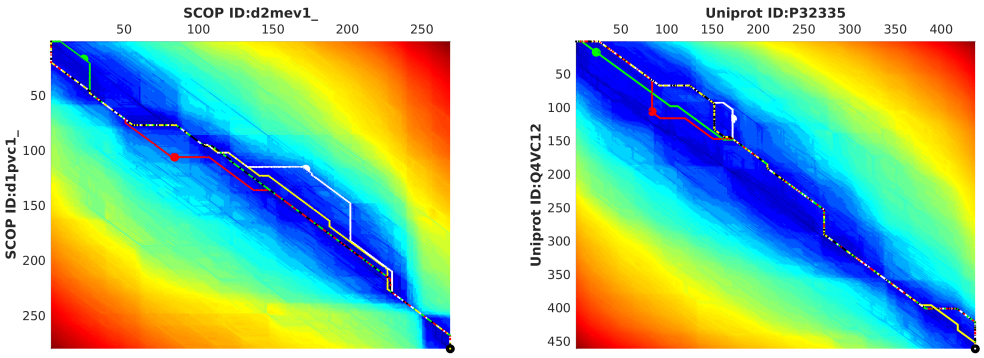
A cell (i, j) in the optimal alignment landscape matrix $\mathcal{L}_{\text{Opt}}$ gives the information content of the best alignment passing through (i, j). This is computed as the combination of the optimal alignment of the prefixes $\langle \mathbf{S}_{1:i}, \mathbf{T}_{1:j} \rangle$ and the suffixes $\langle \mathbf{S}_{i+1:|\mathbf{S}|}, \mathbf{T}_{j+1:\mathbf{T}} \rangle$).

Let the DP history matrices of *forward* alignment and *backward* alignment be: $\{$ $\text{Hist}_{\texttt{m-fwd}}$, $\text{Hist}_{\texttt{i-fwd}}$, $\text{Hist}_{\texttt{d-fwd}}\}$ and $\{$ $\text{Hist}_{\texttt{m-bwd}}$, $\text{Hist}_{\texttt{i-bwd}}$,$\text{Hist}_{\texttt{d-bwd}}\}$, respectively. When joining two equivalent cells (call it (i, j)) in the histories of forward and backward DP computation, the set of all possible state transitions (i.e. the Cartesian product of the vector {`m`, `i`, `d`} and itself) should be taken into consideration when selecting the optimal alignment that passes through cell (i, j). Let $x \rightarrow y$ denote a state transition $\forall \texttt{x}, \texttt{y} \in \{\texttt{m},\texttt{i},\texttt{d}\}$. Then, for $i \neq |\mathbf{S}|$ and $j \neq |\mathbf{T}|$, the total message length of optimal prefix and suffix alignment that passes through cell (i, j) is:

$$\mathcal{L}_{\text{Opt}}(i,j) = \min_{\forall \texttt{x},\texttt{y} \in \{\texttt{m},\texttt{i},\texttt{d}\}} \left\{ \text{Hist}_{\texttt{x-fwd}}(i,j) + \text{Hist}_{\texttt{y-bwd}}(|\mathbf{S}|-i,|\mathbf{T}|-j) + I(\texttt{y}|\texttt{x}) + \log_2\left(\tfrac{1}{3}\right) \right\} \text{ bits}$$

Note: The $\frac{1}{3}$ term is added to avoid the redundant length addition of the initial contexts in both directions. The bottom-most right cell $\mathcal{L}_{\text{Opt}}(|\mathbf{S}|, |\mathbf{T}|)$ is a special case of the above equation, where the $I(\texttt{y}|\texttt{x}) + \log_2\left(\frac{1}{3}\right)$ part can be excluded.

Upon filling all the cells in $\mathcal{L}$, the optimal alignment path will emerge in the landscape as the path of cells having the minimum message length. See Figure 4.2 for examples of optimal alignment landscape visualisations. (Note: The color map ranges from dark blue (= alignments with smaller total message lengths) to dark red (= alignments with higher total message lengths)).



(a) (b)

Figure 4.3: Competing marginal probability landscapes of (a) *Mengo encephalomyocarditis* virus coat protein domain (SCOP ID:`d2mev1_`) and *poliovirus* type 3, sabin strain protein domain (SCOP ID:`d1pvc1_`); (b) Mitochondrial Protein MSS51 of *Saccharomyces cerevisiae* (UniProt ID: P32335) and mitochondrial Putative protein MSS51 homolog of *Homo sapiens* (UniProt ID:Q4VC12).

4.3.2 Marginal Probability Landscape

A cell (i, j) in the marginal probability landscape matrix $\mathcal{L}_{\text{Marginal}}$ gives the negative logarithm of the total probability of all alignments passing through that cell. A consistent combination of *forward* and *backward* DP history matrices computed under the *marginal probability model* results in the construction of $\mathcal{L}_{\text{Marginal}}$.

Let the DP history matrices of *forward* alignment and *backward* alignment be: $\{\text{Tot}_{\texttt{m-fwd}}, \text{Tot}_{\texttt{i-fwd}}, \text{Tot}_{\texttt{d-fwd}}\}$ and $\{\text{Tot}_{\texttt{m-bwd}}, \text{Tot}_{\texttt{i-bwd}}, \text{Tot}_{\texttt{d-bwd}}\}$, respectively. Then, for $i \neq |\mathbf{S}|$ and $j \neq |\mathbf{T}|$, the total message length of all prefix and suffix alignments that passes through cell (i, j) is:

$$\mathcal{L}_{\text{Marginal}}\ (i,j) = \text{-} \underset{\forall \texttt{x},\texttt{y}\ \in\ \{\texttt{m},\texttt{i},\texttt{d}\}}{\texttt{LSE}} \left\{ \text{Tot}_{\texttt{x-fwd}}(i,j) + \text{Tot}_{\texttt{y-bwd}}(|\mathbf{S}|-i, |\mathbf{T}|-j) + I(\texttt{y}|\texttt{x}) + \log_2\left(\tfrac{1}{3}\right) \right\} \text{ bits}$$

Again, $\mathcal{L}_{\text{Marginal}}(|\mathbf{S}|, |\mathbf{T}|)$ is computed by excluding the $I(\texttt{y}|\texttt{x}) + \log_2\left(\frac{1}{3}\right)$ part.

See Figure 4.3 for examples visualisations of marginal probability landscapes. More examples are illustrated in Figure 5.7 of the next chapter, where the *marginal probability model* is evaluated over two benchmark datasets in §5.4.

With this section, the description of the key elements of MML *optimal alignment model* and *marginal probability model* ends here. Next section moves into the details of inferring the *null model*, which is essential to assess the statistical significance of the aforementioned models via the *compression statistic*. It concludes the write-up about the core of the MML protein sequence alignment framework contributed by this thesis.

4.4 The *Null Model* for Protein Sequences

Let us now lay the ground for estimating a *null model* of protein sequences. As described in §3.3, a *null model* explains the observed data (in here, the amino acid sequences) *as is*, without any support of an "interesting" hypothesis. The observed repertoire of proteins across varying species (proteomes) provides the dataset to estimate such a model. The null model encoding

length of any pair of sequences establishes a baseline to test the statistical significance of any alignment hypothesis, as already explored in the previous section of this chapter.

The simplest model of protein sequences is the uniform model, where each amino acid has a probability of $\frac{1}{20}$, taking $-\log_2(\frac{1}{20}) = 4.321$ bits to state. However, amino acid distribution is *non-uniform*, so could be modelled using Markov models.Below section discusses the estimation of any n^{th}-order Markov model ($n \geq 0$) over amino acids, using the MML87 estimator (described in §3.3.3).

4.4.1 MML Estimation of an n^{th}-order Markov Model for Amino Acids

Define an amino acid sequence $S = (S_1 S_2 .. S_{|S|})$ over the amino acid alphabet $\aleph$ (of size k).[4] In an n^{th}-order Markov model, any amino acid $a = S_i \in \aleph$ in the sequence is encoded using a probability of θ_a that is *conditional* on the previous context $(S_{i-n} S_{i-n+1} \cdots S_{i-1})$ of size n.

$$\theta_a = \Pr(S_i \,|(S_{i-n} S_{i-n+1} \cdots S_{i-1}))$$

This section specifically studies the estimates for $n \in \{0, 1, 2, 3\}$. Using the general MML87 estimate derived in Equation 3.3.3, the probability of each amino acid a conditioned on each possible context (denoted by context) is given by:

$$\Pr(a|\text{context}) = \left(\frac{x_{a|\text{context}} + 0.5}{\sum_{\forall b \in \aleph} x_{b|\text{context}} + \frac{k}{2}} \right)$$

where $x_{a|\text{context}}$ is the number of observations of amino acid a with the specified context preceding it. Thus, any sequence S can be encoded using an n^{th}-order Markov model. The order n null model encoding length of the sequence S, denoted by $I_{\text{NULL}}(S)$ is given by:

$$I_{\text{NULL}}(S) \;=\; I(|S|) + \log_2(k^n) + \sum_{i=n}^{|S|} I(S_i|(S_{i-n} S_{i-n+1} \cdots S_{i-1})) \tag{4.12}$$

Here, $I(|S|)$ term refers to the part which conveys the length of S over an integer encoding scheme. The second term accounts for the initial context $S_{0:n-1}$ encoded over a uniform model with $\frac{1}{k^n}$ probability for each context. The summation term refers to the encoding of all amino acid symbols using the MML87 probability estimates inferred over some proteome dataset.

4.4.2 Estimation across Different Proteomes

This section examines the individual MML estimates for a *null model* distribution of amino acids derived over proteomes from a diverse set of organisms, covering the three domains of life (Eukaryota, Bacteria and Archea) and a few viruses (See Appendix A for their information). They also account for both unbiased and biased genomes in terms of their DNA nucleotide composition.

[4] There are are 20 naturally occurring amino acid residues. In addition, there are also five, unnatural amino acid symbols: **B** - Asx (Asp/Asn); **O** - Pyrrolysine ; **U** - Selenocysteine; **X** - any residue; **Z** - Glx (Glu/Gln).

Data curation: Prior to inference, the non-viral proteome sequences were filtered on four different criteria: (1) unreviewed proteins (2) gene unknown proteins, (3) unreliable proteins (predicted/uncertain), and (4) gene multiplicity. Unreliable proteins were identified based on the evidence code. The longest sequence was taken to be the canonical protein for entries under the same gene. An exception to filter (1) was made for *P. ovale wallikeri* as all its proteins appear to be unreviewed. The basic information (including total protein counts and average sequence lengths) before and after filtering are given in Table 4.2. For non-viral proteomes, filtering was not applied due to insufficient numbers of reviewed proteins (Note: none of them had unknown genes). See Table 4.3.

MML message formulation: Each proteome was evaluated in terms of $I(\vec{\Theta}, D)$, the total message length of communicating the Markov model parameter vector $\vec{\Theta}$ and all sequence data D, according to Equation 3.14. Here, $\vec{\Theta} = \{\vec{\theta}_1, \vec{\theta}_2, ..\vec{\theta_{k^n}}\}$, where $\vec{\theta}_i$ represents a point in a unit $(k-1)$-simplex for each possible context c_i in the space $\mathcal{C}$ of all k^n possible contexts. As a result,

$$I(\vec{\Theta}, D) = \sum_{\forall c_i \in \mathcal{C}} \left\{ \frac{d}{2} \log_2 (\kappa_d) - \log_2 [h(\vec{\theta}_i)] + \frac{1}{2} \log_2 [det[Fisher(\vec{\theta}_i)]] + \mathcal{L}(\vec{\theta}_i) + \frac{d}{2} \right\} \tag{4.13}$$

$$= \underbrace{\frac{k^n d}{2} \log_2 (\kappa_d) - k^n \log_2 [h(\vec{\theta}_i)] + \frac{1}{2} \sum_{\forall c_i \in \mathcal{C}} \log_2 [det[Fisher(\vec{\theta}_i)]]}_{\text{First part}} + \underbrace{\mathcal{L}(\vec{\Theta}) + \frac{k^n d}{2}}_{\text{Second part}} \tag{4.14}$$

where $d = k - 1$ is the number of free parameters for each $\vec{\theta}_i \in \vec{\Theta}$, and $h(\vec{\theta}_i)$ is the uniform prior computed using Equation 3.22. (Note: $h(\vec{\theta}_i) = \frac{d!}{\sqrt{d+1}} = \frac{24!}{\sqrt{25}}$ for the $k = 25$ case which takes all natural and unnatural amino acid symbols into account. This reflects the size of the

Table 4.2: Statistics of proteome data across thirteen species covering the three domains of life. (* Filter 1 is ignored for *P. ovale wallikery*)

			Filtering criteria				After filtering	
Species	**Total protein count**	**Average sequence length**	**Unreviewed protein count**	**Gene unknown protein count**	**Unreliable protein count**	**Protein count after multiplicity removal**	**Average sequence length**	**Total residue count**
Eukaryota								
H. Sapiens	70946	336.89	50761	146	621	19261	575.03	11,075,733
A. thaliana	39228	423.45	24032	0	602	14381	459.80	6,612,376
M. musculus	50936	428.74	34082	407	43	16348	573.77	9,379,927
D. melanogaster	21977	681.52	18544	0	8	3385	605.11	2,048,296
S. cerevisiae	6816	445.50	95	0	1086	5635	506.70	2,855,279
P. ovale wallikeri	8636	511.01	8636*	0	7993	643	546.36	351,310
Bacteria								
E. coli	4306	314.96	0	0	622	3684	326.19	1,201,676
C. tetani	2356	336.55	2024	0	0	331	345.79	114,458
M. tuberculosis	3993	333.50	1823	0	100	2066	352.66	728,590
B. subtilis	4260	289.77	72	0	1048	3139	330.04	1,036,009
S. lactis	2225	295.26	1688	0	4	533	336.19	179,188
S. coelicolor	7731	329.65	6943	0	24	764	342.64	261,774
Archea								
M. jannaschii	1787	282.68	0	0	719	1065	311.35	331592

Table 4.3: Statistics of proteome data across five viruses (Note: no filtering is applied)

Species	Total protein count	Unreviewed protein count	Unreliable protein count	Average sequence length	Total residue count
Paramecium bursaria Chlorella virus 1 (PBCV-1)	794	778	1	172.38	136,866
Bacillus virus G	675	675	0	214.76	144,966
Cafeteria roenbergensis virus (CroV)	544	544	0	340.01	184,963
Emiliania huxleyi virus (EhV-86)	472	472	0	260.60	123,003
HumanSARS coronavirus (Human SARS-CoV)	15	0	3	958.47	14,377

Table 4.4: $I(\vec{\Theta}, D)$ total message length breakdown for Markov models of varying order across proteomes of different species and viruses

	Markov order 0		Markov order 1		Markov order 2		Markov order 3	
Species	$I(\vec{\Theta})$	$I(D\|\vec{\Theta})$	$I(\vec{\Theta})$	$I(D\|\vec{\Theta})$	$I(\vec{\Theta})$	$I(D\|\vec{\Theta})$	$I(\vec{\Theta})$	$I(D\|\vec{\Theta})$
H. Sapiens	259.44	46271904.64	3442.93	46103056.53	26911.74	45944536.46	N/A	45797725.06
A. thaliana	251.79	27625159.03	3113.66	27544321.69	20663.95	27482285.38	N/A	27473557.02
M. musculus	251.82	39174068.86	3567.23	39042369.15	26711.70	38951728.55	N/A	38955147.99
D. melanogaster	223.57	8578605.71	2798.67	8536662.45	12047.94	8502019.12	N/A	8629791.19
S. cerevisiae	234.55	11881613.69	2758.86	11850707.41	13413.67	11824212.00	N/A	11891299.77
P. ovale wallikeri	188.38	1455234.29	2000.92	1448584.97	N/A	1451012.63	N/A	1639014.26
E. coli	214.17	4991227.15	2535.25	4973712.71	6505.64	4971551.29	N/A	5134408.20
C. tetani	168.36	467916.57	1401.89	466706.24	N/A	471847.09	N/A	693901.80
M. tuberculosis	207.24	2938021.08	2177.83	2926968.53	1083.20	2925737.91	N/A	3104539.16
B. subtilis	212.84	4296343.93	2391.21	4280574.68	4844.87	4279332.16	N/A	4449041.68
S. lactis	177.63	737782.00	1592.54	735666.49	N/A	740244.71	N/A	951853.89
S. coelicolor	186.02	1053660.10	1747.83	1050625.79	N/A	1054643.80	N/A	1261964.11
M. jannaschii	186.66	1349603.93	2029.27	1343945.63	N/A	1344662.99	N/A	1539994.39
PBCV-1	170.56	574242.32	1493.77	572009.51	N/A	574939.35	N/A	778333.07
Bacillus virus G	172.66	598528.95	1509.66	597093.95	N/A	601864.91	N/A	815506.00
CroV	178.23	752675.55	1607.04	748481.31	N/A	747915.19	N/A	952911.79
EhV-86	168.20	517535.97	1451.99	513214.96	N/A	515656.94	N/A	723015.44
Human SARS-CoV	123.36	60312.63	560.02	60290.94	N/A	64900.47	N/A	312877.08

enormous parameter space for an $\mathbb{L}1$-normalised vector with 24 free dimensions). The total message length $I(\vec{\Theta}, D)$ breakdown of $I(\vec{\Theta})$ and $I(D|\vec{\Theta})$ for each proteome is given in Table 4.4. (Note: all results presented in this section refer to the case of $k = 25$ which includes all natural (20) amino acids and unnatural (5) amino acids. However, this thesis only deals with 20-state amino acid distributions and thus, any inferred *null model* distribution under $k = 25$ is always normalised to derive estimates just for the 20 natural residues.)

In addition, those proteomes were also encoded under the adaptive coding scheme described in §3.2.4. Since it dynamically updates probability models simultaneously at sender and receiver sides, there is no cost associated with sending the model parameters (compared to the first part in Equation 4.13). Thus we expect `|adaptive(message)|` $<$ `|mml(message)|`. However, we can observe no great difference between the two schemes in terms of their *entropy* (bits per residue) values. Table 4.5 presents entropies for each proteome under both the adaptive code and MML code.

Discussion: As n increases, the number of free parameters exponentially increases (i.e. 24, 600 , 15,000 and 375,000 parameters to be estimated for $n = 0, 1, 2, 3$ cases, respectively). This results in the need for a number of data points that is much higher in several magnitudes of order. Otherwise the MML87 approximation does not work well as explained in §3.3.2. N/A

Table 4.5: Entropy (bits per residue) of adaptive codes and MML codes for Markov models of varying order inferred across proteomes of different species and viruses

Species	Adaptive code				MML code			
	n=0	n=1	n=2	n=3	n=0	n=1	n=2	n=3
H. Sapiens	4.17779667	4.16287212	4.15252484	4.16970820	4.17779700	4.16283956	4.15064612	N/A
A. thaliana	4.17783370	4.16611206	4.16273941	4.20085520	4.17783423	4.16604188	4.15931419	N/A
M. musculus	4.17639891	4.16273344	4.15761519	4.19118068	4.17639931	4.16271218	4.15551637	N/A
D. melanogaster	4.18827434	4.16920025	4.16717989	4.27552141	4.18827615	4.16905619	4.15665854	N/A
S. cerevisiae	4.16136032	4.15158392	4.15381432	4.22470692	4.16136155	4.15142137	4.14587355	N/A
P. ovale wallikeri	4.14283417	4.13012640	4.18402795	4.41554733	4.14284443	4.12907655	N/A	N/A
E. coli	4.15373001	4.14136619	4.16128497	4.31525857	4.15373305	4.14108958	4.14259495	N/A
C. tetani	4.08954750	4.09385450	4.20956282	4.498144193	4.08957810	4.08978079	N/A	N/A
M. tuberculosis	4.03275480	4.02093181	4.04845672	4.22445199	4.03275961	4.02029448	4.01710305	N/A
B. subtilis	4.14721610	4.13449391	4.15713082	4.32240698	4.14721955	4.13410105	4.13527009	N/A
S. lactis	4.11833448	4.11704233	4.20542581	4.48685995	4.11835404	4.11444417	N/A	N/A
S. coelicolor	4.02577264	4.02193717	4.08898817	4.34268726	4.02578603	4.02016095	N/A	N/A
M. jannaschii	4.07062562	4.06000245	4.10896192	4.33855221	4.07063677	4.05912961	N/A	N/A
PBCV-1	4.19687421	4.19365411	4.28822487	4.51069295	4.19689980	4.19025381	N/A	N/A
Bacillus virus G	4.12992087	4.13248044	4.23425136	4.50933939	4.12994504	4.12926900	N/A	N/A
CroV	4.07027499	4.05785806	4.11650459	4.35144766	4.07029393	4.05534269	N/A	N/A
EhV-86	4.20884583	4.18796662	4.28328290	4.51012196	4.20887430	4.18418209	N/A	N/A
Human SARS-CoV	4.20341583	4.26565712	4.40951156	4.36520923	4.20365796	4.23252123	N/A	N/A

entries in the result tables are evident of such cases, where negative first part message lengths ($I(\vec{\Theta})$) are observed. For some proteomes (including all viral proteomes),even $n = 2$ models give invalid total message lengths due to insufficient amount of data. Table 4.6 gives an idea into how entropies differ across varying orders, by computing the bits per residue in terms of the negative log likelihood.

4.4.3 Inference of a *Null Model* over the Proteins from All Species

While a species-wise *null model* is beneficial for encoding protein sequences from that particular species, a more universal *null model* inferred over all observed proteins across all different species is ideal for general applicability. UniProt (UniProt Consortium and others, 2017) is a suitable data source for this purpose. The full reviewed database of UniProt had 554,515 protein sequences in total, with an average length of 357.99 residues (at the time of this experiment). After filtering for reliable data (22,599 gene unknown proteins, 14,314 unreliable proteins) and removing multiplicity, 129,788 sequences were left with an average length of 454.30 and 58,963,216 residues. Table 4.7 and 4.8 present $I(\vec{\Theta}, D)$ total message length breakdown and entropies for Markov models of varying order. As expected, entropy decreases when the Markov order n increases.

4.4.4 A Short Analysis of *Null Models* across Different Species

A short analysis of *null model* estimates across different species and the UniProt can provide insights on their similarities and differences with respect to one another. Figure 4.4 illustrates a KL divergence distance matrix across all inferred *null models* alongside a dendrogram describing an agglomerative hierarchical clustering (with average linkage) of those models based on the

Table 4.6: Entropy (bits per residue) in terms of the negative log likelihood ($I(\vec{\Theta}|D)$) for Markov models of varying order inferred across proteomes of different species and viruses

Species	**n = 0**	**n = 1**	**n = 2**	**n = 3**
H. Sapiens	4.17777201	4.16248963	4.14723940	4.11053785
A. thaliana	4.17779354	4.16550554	4.15455279	4.11396020
M. musculus	4.17637062	4.16228573	4.15151507	4.12419443
D. melanogaster	4.18815855	4.16747855	4.14549406	4.08109271
S. cerevisiae	4.16127334	4.15030356	4.13738615	4.06993308
P. ovale wallikeri	4.14225892	4.12214898	4.09949165	3.89544545
E. coli	4.15354042	4.13861964	4.12817687	4.04759926
C. tetani	4.08795592	4.07375131	4.02791311	3.69914278
M. tuberculosis	4.03245141	4.01671134	4.00076545	3.88975122
B. subtilis	4.14699739	4.13137518	4.12014948	4.03330122
S. lactis	4.11726613	4.10314127	4.07072178	3.80242301
S. coelicolor	4.02500930	4.01183073	3.98749908	3.78746091
M. jannaschii	4.07002165	4.05170458	4.02254210	3.82846711
PBCV-1	4.19552710	4.17617742	4.12168938	3.71040108
Bacillus virus G	4.12863455	4.11586954	4.07712637	3.75950689
CroV	4.06923676	4.04431430	3.98509421	3.68942153
EhV-86	4.20736614	4.16885894	4.10426355	3.67885431
Human SARS-CoV	4.19387372	4.16346452	3.76158155	2.94719074

Table 4.7: $I(\vec{\Theta}, D)$ total message length breakdown for the Markov models of varying order, inferred over all reliable UniProt protein sequences

Markov order 0		**Markov order 1**		**Markov order 2**		**Markov order 3**	
$I(\vec{\Theta})$	$I(D\|\vec{\Theta})$	$I(\vec{\Theta})$	$I(D\|\vec{\Theta})$	$I(\vec{\Theta})$	$I(D\|\vec{\Theta})$	$I(\vec{\Theta})$	$I(D\|\vec{\Theta})$
282.80	246608016.52	4564.68	245951600.02	47844.93	245345939.72	N/A	244434825.28

Table 4.8: Entropy (bits per residue) for the Markov models of varying order, inferred over all reliable UniProt protein sequences

Scheme	**Markov order 0**	**Markov order 1**	**Markov order 2**	**Markov order 3**
Adaptive code	4.18240917	4.17134914	4.16205736	4.15673963
MML	4.18240924	4.17134921	4.16181140	N/A
MML-$\mathcal{L}(\vec{\Theta})$	4.18240415	4.17126446	4.16081646	4.14096002

given distance matrix. Following intuitive and interesting features can be grasped from this structure.

- *H. sapiens* and its model organism *M. musculus* are the closest, reflecting the genomic similarity between humans and house mouse. Looking further, all other model organisms of humans (i.e. *D. melanogaster*, *A. thaliana* and *E. coli* – except for *S. cerevisiae*) fall under a single group at a higher level. *S. cerevisiae* is somewhat related to the above group at a much higher level, yet interestingly being grouped with three viral proteomes: *Paramecium bursaria Chlorella virus 1*, *Emiliania huxleyi virus* and *Human SARS coronavirus*.

- Species that have compositional bias in their genomes are clustered together. For instance, both prokaryotes, *M. tuberculosis* and *S. coelicolor* with GC rich genomes are grouped together. Another group is *C. tetani* and *M. jannaschii* which are closer to each other.

C. tetani is an AT rich genome. *M. jannaschii* genome is also lower in GC content on average (Garrett, 1996). However, the AT rich *L. lactis* resides in a different group. (Note: compositional bias of these genomes are discussed by Yu et al. (2003)).

- Another observation is that, all multi-cellular eukaryotic proteomes are in a single cluster. The single-cellular *P. ovale wallikeri* is separately positioned.
- *B. subtilis* and *L. lactis* are both gram positive bacteria, and they are grouped together.
- *Cafeteria roenbergensis virus* is a large virus (Fischer et al., 2010), separated from many other proteomes – specifically from other viruses. All other viruses are in the same high-level group, except for this virus and *Bacillus virus G*.
- An interesting group is *Bacillus virus G* (a bacteriophage virus) and *P. ovale wallikeri* (a eukaryote)

Overall, we can conclude that UniProt estimates are good enough to represent eukaryotes, prokaryotes and viruses in general, despite exceptions to the ones with biased genomes (AT rich/GT rich) or having other interesting features (such as, being a large virus or a bacteriophage). Figure 4.5 presents *null model* probability distributions of all species and UniProt protein sequences involved in this study.

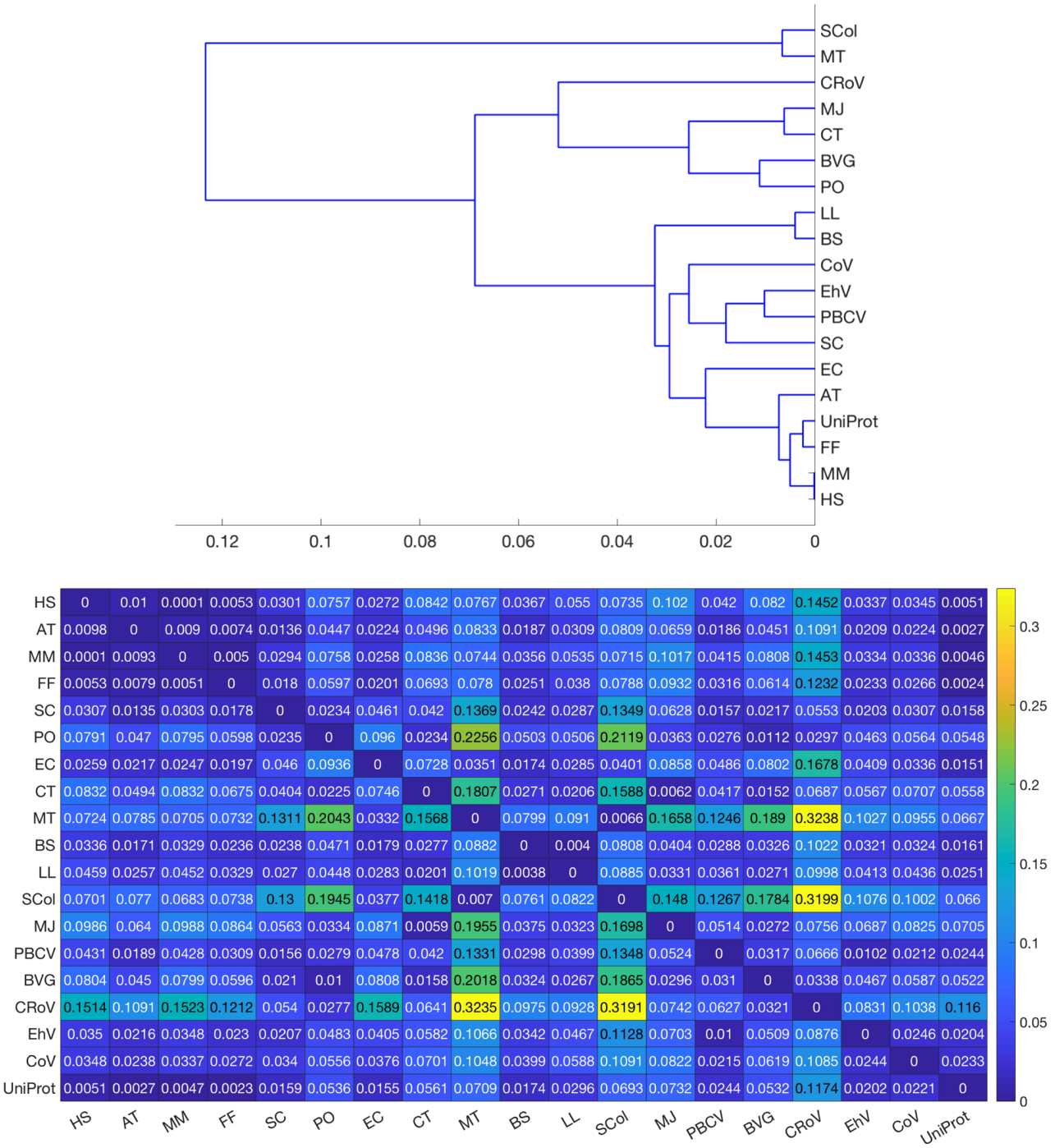
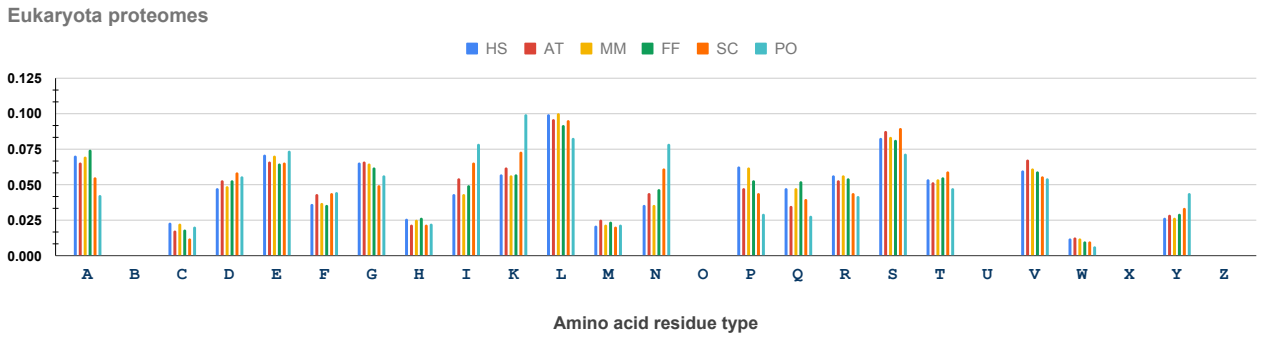
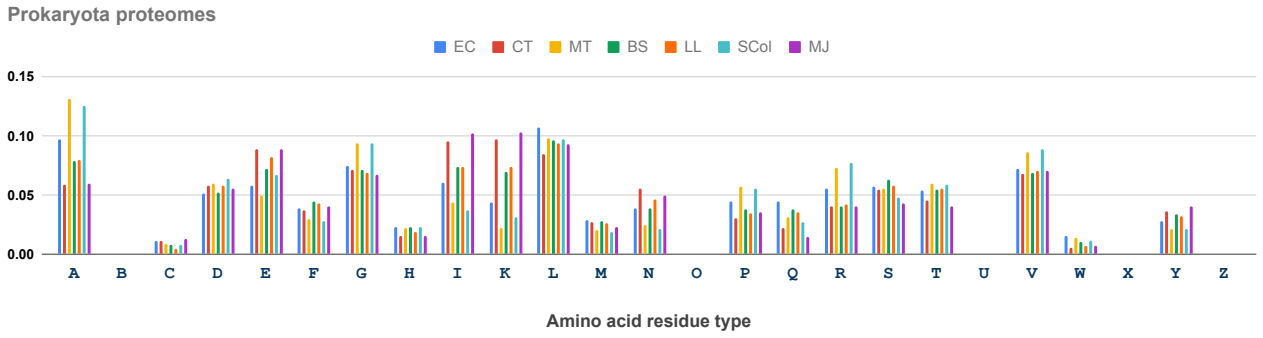


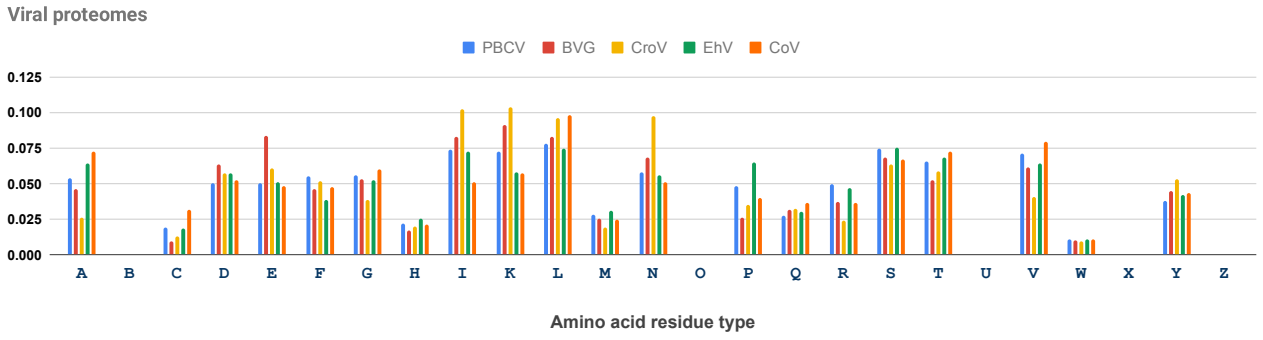
Figure 4.4: KL divergence distance matrix and clustering of Markov order 0 *null models* across the proteomes of **Eukaryota:** *H. sapiens* (HS), *A. thaliana* (AT), *M. musculus* (MM), *D. melanogaster* (FF), *S. cerevisiae* (SC), *P. ovale wallikeri* (PO); **Prokaryota:** *E. coli* (EC), *C. tetani* (CT), *M. tuberculosis* (MT), *B. subtilis* (BS), *L. lactis* (LL), *S. coelicolor* (SCol), *M. jannaschii* (MJ); **Viruses:** *Paramecium bursaria Chlorella virus 1* (PBCV), *Bacillus virus G* (BVG), *Cafeteria roenbergensis virus* (CroV), *Emiliania huxleyi virus* (EhV), *Human SARS coronavirus* (CoV); and **UniProt** protein sequences. Note: all distance measures are in nits. (Note: clustering was generated using MATLAB R2017a (The Mathworks, Inc., 2017))



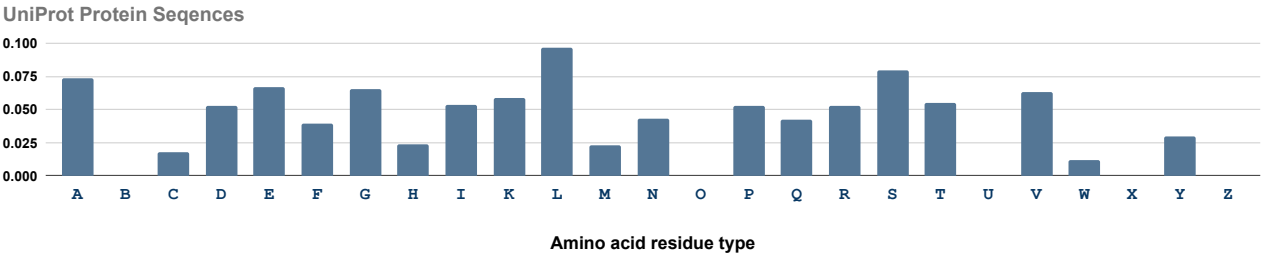
(a)



(b)



(c)



(d)

Figure 4.5: MML estimates of amino acid *null model* probabilities (reflecting the residue frequencies), (a) for proteomes across **Eukaryota:** *H. sapiens* (HS), *A. thaliana* (AT), *M. musculus* (MM), *D. melanogaster* (FF), *S. cerevisiae* (SC), *P. ovale wallikeri* (PO); (b) for proteomes across **Prokaryota:** *E. coli* (EC), *C. tetani* (CT), *M. tuberculosis* (MT), *B. subtilis* (BS), *L. lactis* (LL), *S. coelicolor* (SCol), *M. jannaschii* (MJ); (c) for **Viral proteomes:** *Paramecium bursaria Chlorella virus 1* (PBCV), *Bacillus virus G* (BVG), *Cafeteria roenbergensis virus* (CroV), *Emiliania huxleyi virus* (EhV), *Human SARS coronavirus* (CoV); and (d) for the collection of **UniProt** protein sequences.

Chapter 5

Alignment Three-state Machine as a Function of Evolutionary Time

"*Nothing in Biology Makes Sense Except in the Light of Evolution*"

– Theodosius Dobzhansky

The previous chapter established the core MML framework for protein sequence comparison. However, it mainly estimated the best alignment three-state machine parameters $\vec{\Theta}$ based on a bland (uniform) prior, although it provided a scope for the use of more informative ones. Thus, this chapter achieves the goal of estimating informative priors for the three-state machine as a function of evolutionary time. Specifically, it discusses the derivation of an evolutionary-time-dependent three-state machine *given* any stochastic model of amino acid substitutions. Addressing this relationship unifies the treatment of substitutions, insertions and deletions, which remain disconnected in the practice of sequence comparison. Specifically, the outcome of this chapter is, for any given Markov model of amino acid substitution, a corresponding set of (Markov-)time-dependent Dirichlet probability distributions inferred using MML over any stated source collection of alignments.

Parts of the material presented in this chapter are published in:
Sumanaweera, D., Allison, L. and Konagurthu, A.S., 2019. Statistical compression of protein sequences and inference of marginal probability landscapes over competing alignments using finite state models and Dirichlet priors. Bioinformatics, 35(14), pp.i360-i369.
DOI: 10.1093/bioinformatics/btz368

5.1 Motivation

The relationship between evolutionary time and indel events has received a limited attention when studying the parameter space of a protein sequence alignment. Several noticeable efforts are present in literature, to connect indel events with sequence divergence. Amongst them is Blake and Cohen (2001) who tested the improvement of alignments by constructing a set of substitution matrices[1] for different contexts of evolutionary time and obtained optimal

[1] structural superposition based substitution matrices

gap penalties for each one. They estimated matrices for defined ranges of sequence identity percentage, binned at a width of ten, while also obtaining versions for aggregated bins. Earlier, a comprehensive study was done by Vogt et al. (1995) to optimise gap penalties for each of a diverse set of scoring matrices[2] under a range of sequence identity percentage bins, gauging the improvement of sequence alignments in comparison to a reliable structural alignment dataset.

Several papers (Gonnet et al., 1992; Chang and Benner, 2004; Benner et al., 1993) have also explored the relationship between evolutionary time[3] and the length of gaps. They challenged the common notion of gap length modelling under the geometric distribution, by empirically estimating a generalised Zipfian distribution. Under a Zipfian model, Pr(gap of length L)= $c_1.L^{-c_2}$, where c_1 and c_2 are parameters. (Chang and Benner (2004) empirically estimated $c_1 = 1, c_2 = 1.8$. This was previously estimated over a small dataset as $c_1 = 1, c_2 = 1.7$). Gonnet et al. (1992) and Benner et al. (1993) observed that the occurrence probability of a gap grows in a linear fashion with evolutionary time t, for small values of t under the PAM Markov model of evolution (PAM-t). However, they claim the Zipfian distribution parameters to be independent of time. Chang and Benner (2004) further verified that the Zipfian approximation does not change across a few bins spanned over PAM-10 and PAM-100, and also over the f2 measure ("the fraction of conserved nucleotides at the third position of the corresponding codon where the residue is conserved"), accounting for varying levels of selective pressure. A surprising observation is a moderate decrease in expected gap length with respect to an increase in PAM-t time (Benner et al., 1993). On the contrary, experiments by Pascarella and Argos (1992) have evinced an exponentially decreasing behaviour of the expected gap length as the percentage of sequence identity increases. Qian and Goldstein (2001) presented another empirical indel length distribution by fitting a linear combination of four exponential functions (multi-exponential distribution) to explain the probability of a gap over a set of distantly related proteins of $< 25\%$ sequence identity. However it does not explore the distribution as a function of time. Later, Pang et al. (2005) parameterised this on varying evolutionary time. Recently, Holmes (2017) discussed approaches for time-dependent gap length modelling in pairwise alignments, highlighting the potential of applying continuous-time Markov models for the purpose.

Despite all these efforts, the field has resisted formal attempts to connect indels with sequence divergence. In practice, the affine gap penalty function with empirically chosen gap open and extension penalties are considered a good approximation to address insertions and deletions, under the assumption that the gap lengths follow a geometric distribution (Cartwright, 2006). While this is a useful approximation in practice, it however does not address the problem. Therefore, this chapter is an attempt to address this disconnect using rigorous probabilistic models.

5.2 Estimation of Dirichlet Distributions

Recalling the symmetric properties of the three-state machine discussed in §4.1.1, it entails a point estimate in a unit 1-simplex (for the case of Pr(m|m)) and unit 2-simplex (for the case of Pr(i|i) and Pr(m|i)). Accordingly, the free-parameter vector of the three-state machine is: $\vec{\Theta} = \{\Pr(\mathtt{m}|\mathtt{m}), \Pr(\mathtt{i}|\mathtt{i}), \Pr(\mathtt{m}|\mathtt{i})\}$. The usage of $\vec{\Theta}$ under predefined Dirichlet probability models was briefly introduced in §4.1.1. Here, we present the details of how such Dirichlet distributions can be derived from any corpus of existing (i.e. benchmark) alignments. Specifically, these

[2]including a series of PAM and BLOSUM scoring matrices, as well as the Gonnet matrix (Gonnet et al., 1992).

[3]defined by time t under the PAM Markov model of evolution. See Chapter 6 for a comprehensive discussion.

Dirichlet distributions are inferred as a function of the evolutionary-time-parameter under a specified Markov matrix of amino acid substitutions. These time-dependent Dirichlets, in turn, allow us to estimate the free-parameters $\vec{\Theta}$ of the 3-state machine as a function of the time-parameter.[4]

More formally, suppose we have a stochastic model $\mathbf{M}$ that generates a set of time-dependent transition-probability matrices of amino acid substitutions, $\Phi = \{\mathbf{M}_1, \mathbf{M}_2, ..., \mathbf{M}_{t_{\text{Max}}}\}$, where the time-parameter is an integer in the range $t \in [1, t_{\text{Max}}]$. Then, given any benchmark collection of protein alignments, $\mathbf{A} = \{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_{|\mathbf{A}|}\}$, the optimal time-parameter t_i can be inferred for each alignment $\mathcal{A}_i \in \mathbf{A}$, using the the search method described in §4.1.2 of the previous chapter. This yields time (t) dependent subsets of alignments, on which Dirichlet distributions and respective 3-state machine parameters are inferred. The next section explains the details for these estimations.

5.2.1 Dirichlet Probability Distribution

The Dirichlet distribution models proportions (ratios), that is, points in a $(k-1)$-dimensional unit simplex. It is often used as a prior to the multi-state distribution for which it is a conjugate prior. The Beta distribution is the $k = 2$ case of this model. Thus, Dirichlet is also spoken of as the generalisation of the Beta distribution. This section briefly describes the associated statistics of the Dirichlet distribution.

Let $\text{Dir}(\vec{\alpha})$ be a Dirichlet distribution with a parameter vector $\vec{\alpha} = [\alpha_1,\alpha_2,...\alpha_k]$. that describes a data sample $\vec{\Theta} = [\theta_1,\theta_2,...\theta_k]$ ($\sum_{i=1}^{\text{k}} \theta_i = 1$), representing a probability vector over some state space of dimension k. Each α_i is analogous to a pseudo-count of successes for the state i. The following reparameterisation of $\vec{\alpha}$ shows how the distribution's probability density concentrates (controlled by the parameter κ) around its mean vector $\vec{\mu}$ within the k$-$1 simplex.

$$\vec{\alpha} = \kappa\vec{\mu} = \underbrace{\left(\sum_{i=1}^{k}\alpha_i\right)}_{\kappa=\text{concentration}} \underbrace{\left[\frac{\alpha_1}{\sum_{i=1}^{k}\alpha_i}, \frac{\alpha_2}{\sum_{i=1}^{k}\alpha_i}, \ldots, \frac{\alpha_k}{\sum_{i=1}^{k}\alpha_i}\right]}_{\vec{\mu}=\text{mean vector}}$$

The mode vector $[x_1, x_2, ...x_{\text{k}}]$ of $\text{Dir}(\vec{\alpha})$ is defined by $x_i = \frac{\alpha_i - 1}{\kappa - \text{k}}$ (only for $\alpha_i > 0$). Following defines the probability density function (PDF) $f(\vec{\Theta}\ |\vec{\alpha})$ of $\text{Dir}(\vec{\alpha})$:

$$f(\vec{\Theta}\ |\vec{\alpha}) = \frac{1}{\text{B}(\vec{\alpha})} \prod_{i=1}^{\text{k}} {\theta_i}^{\alpha_i - 1} \tag{5.1}$$

where, $\text{B}(\vec{\alpha})$ is the multivariate form of the Beta function:

$$\text{B}(\vec{\alpha}) = \frac{\prod_{\forall i}\Gamma(\alpha_i)}{\Gamma(\sum_{\forall i}\alpha_i)}$$

[4] In this thesis, an independent Dirichlet model is estimated for each discrete bin of evolutionary time in some defined time range $[1, t_{\texttt{Max}}]$. However, as a future direction, this modelling exercise can be pursued further by exploring ways to parameterise the three-state machine using continuous-time Markov models.

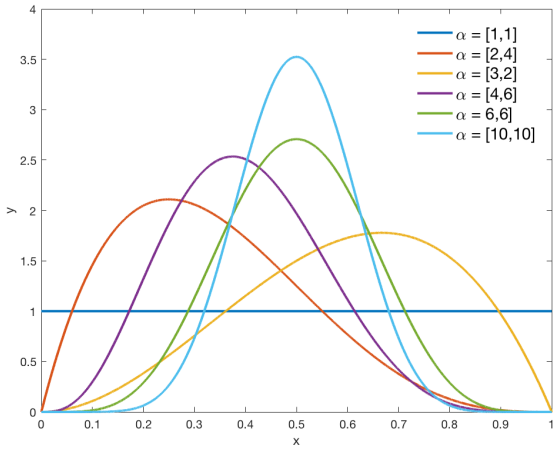
See Figure 5.1 for example PDF plots). Respectively, the likelihood over some data D with N samples: $D = [\vec{\Theta}_1, \vec{\Theta}_2, ...\vec{\Theta}_N]$ is defined as:

$$f(D \mid \vec{\alpha}) = \prod_{n=1}^{\mathrm{N}} f(\vec{\Theta}_n \mid \vec{\alpha}).$$

Accordingly, the negative log likelihood function is:

$$\mathcal{L}(\vec{\alpha}) = -\mathrm{N} \log_2 \Gamma(\kappa) + \mathrm{N} \sum_{i=1}^{\mathrm{k}} \log_2 \Gamma(\alpha_i) - \sum_{n=1}^{\mathrm{N}} \sum_{i=1}^{\mathrm{k}} (\alpha_i - 1) \log_2 (\theta_{n,i})$$

Special case at $k = 2$: Beta distribution



This figure plots a few example visualisations of a unit 1-simplex under different parameter settings for Beta distribution. The x-axis positions the unit 1-simplex while the y-axis gives the probability density value under the PDF of the Dirichlet distribution given in the Equation 3.23. Note how the curve becomes sharper at its mode when concentration ($\kappa = \alpha_1 + \alpha_2$) parameter increases. The shape of the distribution is controlled by the parameter vector $\vec{\alpha}$. When $\alpha_1 = \alpha_2$, it gives a symmetric distribution centered at x= 0.5. $\vec{\alpha}$= [1, 1] case denotes the uniform distribution.

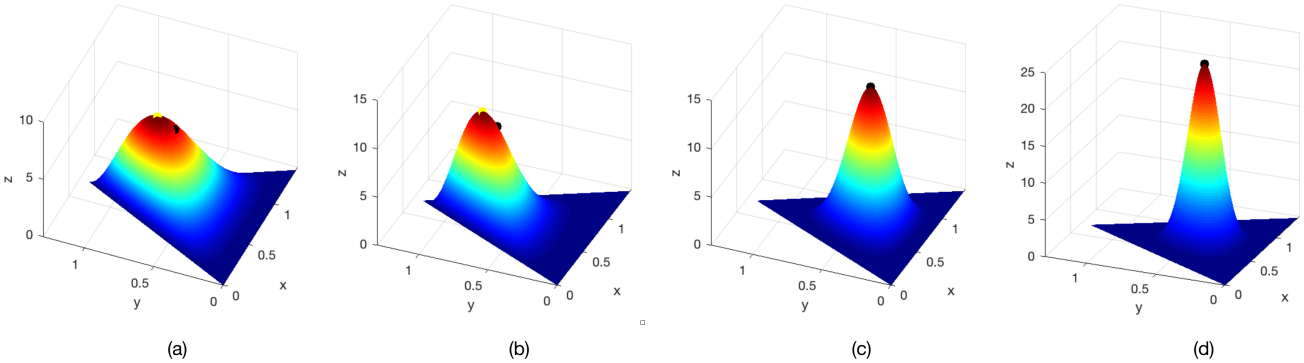


Figure 5.1: Some example visualisations of a unit 2-simplex under four different parameter settings for Dirichlet distribution: (a) $\vec{\alpha} = [2, 2, 4]$, (b) $\vec{\alpha} = [3, 2, 6]$, (c) $\vec{\alpha} = [6, 6, 6]$, and (d) $\vec{\alpha} = [10, 10, 10]$. The x and y axes position the unit simplex, while z-axis gives the Dirichlet PDF. The yellow colour dot and black colour dot refer to the probability density values of the mode vector and mean vector, respectively. Note how the mode and mean get distant as the distribution becomes asymmetric in (a) and (b), while they coincide for symmetric Dirichlet in (c) and (d). Also, as κ (the sum of the pseudocounts) increases, the distribution becomes more and more sharp at the mode vector.

where $\Gamma(\cdot)$ is Euler's Gamma function[5] (Note: $\Gamma(\mathrm{z}) = \int_0^\infty x^{z-1}e^{-x}dx$ definite integral for $z \in \mathrm{C}, \mathrm{Re}z > 0$). Solving for the first partial derivative, we get:

$$\frac{\partial}{\partial \alpha_i}(\mathcal{L}(\vec{\alpha})) = \mathrm{N}\psi_0(\kappa) - \mathrm{N}\psi_0(\alpha_i) + \sum_{n=1}^{\mathrm{N}} \log_2(\theta_{n,i})$$

where $\psi_0(\cdot)$ is the Polygamma function of order 0 (Digamma function) (Note: Polygamma function of order m: $\psi_m(x) = \frac{\partial^{m+1}}{\partial x^{m+1}} \log_2[\Gamma(\mathrm{x})]$). As explained by Allison (2018), the observed Fisher H of the negative log likelihood function is in the below special form:

$$H = \begin{pmatrix} \frac{\partial^2\mathcal{L}}{\partial\alpha_1^2} & \frac{\partial^2\mathcal{L}}{\partial\alpha_1\partial\alpha_2} & \frac{\partial^2\mathcal{L}}{\partial\alpha_1\partial\alpha_3} & .. & .. \\ \frac{\partial^2\mathcal{L}}{\partial\alpha_2\partial\alpha_1} & \frac{\partial^2\mathcal{L}}{\partial\alpha_2^2} & \frac{\partial^2\mathcal{L}}{\partial\alpha_2\partial\alpha_3} & .. & .. \\ ... & & ... & & \\ ... & & & ... & \\ ... & & & & \frac{\partial^2\mathcal{L}}{\partial\alpha_k^2} \end{pmatrix} = \begin{pmatrix} c_1 - z & -z & -z & .. & .. \\ -z & c_2 - z & -z & .. & .. \\ ... & & ... & & \\ ... & & & ... & \\ ... & & & & c_k - z \end{pmatrix}$$

where

$$-z = \frac{\partial^2}{\partial\alpha_i\alpha_j}(\mathcal{L}(\vec{\alpha})) = -N\psi_1[\sum_{i=1}^{k}\alpha_i]$$

$$c_i - z = \frac{\partial^2}{\partial\alpha_i^2}(\mathcal{L}(\vec{\alpha})) = N\psi_1(\alpha_i) - N\psi_1[\sum_{i=1}^{k}\alpha_i]$$

The expected Fisher information matrix $Fisher(\vec{\alpha})$ can be computed as $\mathbb{E}\left[\frac{\partial^2}{\partial\alpha^2}(\mathcal{L}(\vec{\alpha}))\right]$ directly from H, since all associated terms are constants (i.e. they do not depend on the data points). Consequently,

$$Fisher(\vec{\alpha}) = H$$

This yields a closed form for the matrix determinant. Accordingly, the determinant of this Fisher matrix $Fisher(\vec{\alpha})$, which indicates how sensitive the expected negative log likelihood function $\mathcal{L}$ is to the changes of $\vec{\alpha}$, is given by:

$$det[Fisher(\vec{\alpha})] = \mathrm{N}^{\mathrm{k}}\left(\prod_{i=1}^{\mathrm{k}}\psi_1(\alpha_i)\right)\left(1 - \psi_1(\kappa)\sum_{i=1}^{\mathrm{k}}\frac{1}{\psi_1(\alpha_i)}\right) \tag{5.2}$$

where $\psi_1(\cdot)$ is the Polygamma function of order 1 (Trigamma function)

Sampling randomly from a Dirichlet distribution: Suppose a k-dimensional probability vector $\vec{\theta} = (\theta_1, ..., \theta_k)$ is to be sampled from a k-dimensional $\mathrm{Dir}(\vec{\alpha})$. A random sampling is done as follows:

1. For each component $\alpha_i \in \vec{\alpha}$, generate a Gamma distributed random sample y_i from the Gamma distribution[6] $\Gamma(\alpha_i, 1)$

[5] Gamma function extends the factorial function to complex numbers. (And for integer n, $\Gamma(n) = (n-1)!$).

[6] The Gamma distribution is a family of continuous probability distributions including the Exponential distribution, denoted by $\Gamma(k, \theta)$ with a shape parameter k and scale parameter θ. It models the waiting time

2. $\mathbb{L}1$-normalise the sampled vector $\vec{y} = (y_1, ...y_k)$

The resultant vector $\vec{y}'$ with $y'_i = \frac{y_i}{\sum_{i=1}^{k} y_i}$ is Dirichlet distributed under $\text{Dir}(\vec{\alpha})$.

5.2.2 MML based Dirichlet Estimation over Alignments

This section discusses the formulation of an MML protocol to communicate a set of alignment three-state strings $\mathbf{A}$ of size $|\mathbf{A}| = N$ as data. An alignment string $\mathcal{A}_n \in \mathbf{A}$ is encoded using a 3-state machine with parameter $\vec{\Theta}_n = \{\vec{\Theta}_{n(\mathtt{m})} = [\Pr(\mathtt{m}|\mathtt{m})], \vec{\Theta}_{n(\mathtt{i})} = [\Pr(\mathtt{i}|\mathtt{i}), \Pr(\mathtt{m}|\mathtt{i})]\}$, defined over two Dirichlet priors: $\text{Dir}_{\mathtt{m}}(\vec{\alpha}_{\mathtt{m}})$ and $\text{Dir}_{\mathtt{i}}(\vec{\alpha}_{\mathtt{i}})$ (Note: these priors are applied commonly across all alignments in $\mathbf{A}$). The goal is to find the optimal $\vec{\alpha}_{\mathtt{m}}$ and $\vec{\alpha}_{\mathtt{i}}$ that minimise the total message length $I(\vec{\alpha}_{\mathtt{m}}, \vec{\alpha}_{\mathtt{i}}, \vec{\Theta}, \mathbf{A})$, where $\mathbf{\Theta} = \{\vec{\Theta}_1, \vec{\Theta}_2, \ldots, \vec{\Theta}_N\}$. Since the 1-simplex Dirichlet and 2-simplex Dirichlet models are independent from each other (as shown in §4.1.1 with Figure 4.1), this minimisation can be performed separately.

Below presents the general total message formulation which requires the communication of $\vec{\alpha}_{\mathtt{x}}$, $\mathbf{\Theta}_{(\mathtt{x})} = \{\vec{\Theta}_{1(\mathtt{x})}, \vec{\Theta}_{2(\mathtt{x})}, \ldots, \vec{\Theta}_{N(\mathtt{x})}\}$ and $\mathbf{A}_{(\mathtt{x})}$ jointly, for any state $\mathtt{x} \in \{\mathtt{m}, \mathtt{i}\}$. Note: $\mathbf{A}_{(\mathtt{x})}$ contains only the data instances relevant to the state of focus (i.e. for the case of $\mathtt{x} = \mathtt{m}$, all instances of `mm`, `mi`, `md` state transitions are taken together; For the case of $\mathtt{x} = \mathtt{i}$, all instances of `ii`, `im`, `id`, `dd`, `dm`, `di` state transitions are taken together). The associated two part message length is:

$$I(\vec{\alpha}_{\mathtt{x}},\, \mathbf{\Theta}_{(\mathtt{x})},\, \mathbf{A}_{(\mathtt{x})}) = \underbrace{I(\vec{\alpha}_{\mathtt{x}}) + I(\mathbf{\Theta}_{(\mathtt{x})}\ |\vec{\alpha}_{\mathtt{x}})}_{\text{First part}} + \underbrace{I(\mathbf{A}_{(\mathtt{x})}\ |\mathbf{\Theta}_{(\mathtt{x})}, \vec{\alpha}_{\mathtt{x}}\ )}_{\text{Second part}} \quad \text{bits} \tag{5.3}$$

The first part term $I(\vec{\alpha}_{\mathtt{x}})$ denotes the statement cost of the Dirichlet parameters $\vec{\alpha}_{\mathtt{x}}$. Using the Wallace and Freeman (1987) methods of estimation detailed in §3.3.2, this term expands to:

$$I(\vec{\alpha}_{\mathtt{x}}) = -\log_2 [h(\vec{\alpha}_{\mathtt{x}})] + \frac{k}{2} \log_2 (c_k) + \frac{1}{2} \log_2 [det(Fisher(\vec{\alpha}_{\mathtt{x}}))] \quad \text{bits}$$

where $h(\vec{\alpha}_{\mathtt{x}})$ is the prior on the Dirichlet parameters $\vec{\alpha}_{\mathtt{x}}$; c_k is the optimal quantising lattice constant (Conway and Sloane, 1984) associated with k degrees of freedom ($c_2 = \frac{5}{36\sqrt{3}}$ and $c_3 = \frac{19}{192 \times 2^{\frac{1}{3}}}$); and $det[Fisher(\vec{\alpha}_{\mathtt{x}})]$ is the determinant of the Fisher information matrix of $\vec{\alpha}_{\mathtt{x}}$ given by the Equation 5.2. Further, using the reparameterisation of a Dirichlet parameter $\vec{\alpha}$ previously stated in §5.2.1, the prior term can be decomposed as:

$$h(\vec{\alpha}_{\mathtt{x}}) = h(\kappa_{\mathtt{x}}) \cdot h(\vec{\mu}_{\mathtt{x}})$$

by applying independent priors for concentration parameter $\kappa_{\mathtt{x}}$ and the $\mathbb{L}1$-normalised mean vector $\vec{\mu}_{\mathtt{x}}$. In this thesis, $\vec{\mu}_{\mathtt{x}}$ is assumed to be uniformly distributed in a unit $k-1$ simplex. Thus, $h(\vec{\mu}_{\mathtt{x}})$ is simply the reciprocal of the simplex volume. As a result, $h(\vec{\mu}_{\mathtt{m}}) = \frac{1}{\sqrt{2}}$ and $h(\vec{\mu}_{\mathtt{i}}) = \frac{2}{\sqrt{3}}$.

On the other hand, $\kappa_{\mathtt{x}}$ controls the Dirichlet concentration about the distribution's mode. For any y diffusing with k degrees of freedom, Wallace and Dowe (1993) defined a well-behaved prior for y in the form:

$$g(y) = \frac{y^{k-1}}{(1+y^2)^{\frac{k+1}{2}}}$$

until the k^{th} occurrence of some continuous-time valued event. Usually, inverse transform sampling or an acceptance-rejection method (Ahrens and Dieter, 1982) is applied for generating a Gamma distributed variate.

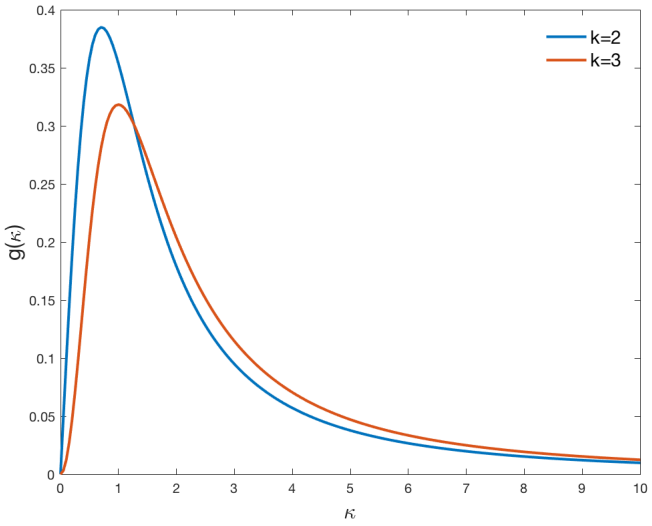


Figure 5.2: Prior density function for 1-simplex ($k = 2$) and 2-simplex ($k = 3$) Dirichlet concentration parameter κ

Before applying $g(\cdot)$ as a prior for $\kappa_{\mathtt{x}}$, a normalisation constant is computed to ensure this as a valid probability distribution:

$$\int_0^\infty g(y) = 1$$

For $k = 2$, we can directly take $h(\kappa_{\mathtt{m}}) = g(\kappa_{\mathtt{m}})$, as the above is satisfied. However for $k = 3$, $\int_0^\infty g(y) = \frac{\pi}{4}$, which is not equal to 1. Therefore, this yields $h(\kappa_{\mathtt{i}}) = \frac{4}{\pi} g(\kappa_{\mathtt{i}})$. Figure 5.2 plots the normalised prior density functions for $\kappa_{\mathtt{m}}(k = 2)$ and $\kappa_{\mathtt{i}}$ $(k = 3)$.

The first part $I(\mathbf{\Theta}_{(\mathtt{x})}|\vec{\alpha}_{\mathtt{x}})$ term relates to sending the parameter vector $\vec{\Theta}_{n(\mathtt{x})}$ of each alignment $\mathcal{A}_n \in \mathbf{A}$ given a Dirichlet prior with parameter $\vec{\alpha}_{\mathtt{x}}$. Using the approximation specified in Equation 3.15, the term can be expanded as:

$$I(\mathbf{\Theta}_{(\mathtt{x})}|\vec{\alpha}_{\mathtt{x}}) = \left\{ \sum_{n=1}^{N} \frac{1}{2} \log_2 \left(1 + \frac{det[Fisher(\vec{\Theta}_{n(\mathtt{x})})] {c_{k-1}}^{k-1}}{f(\vec{\Theta}_{n(\mathtt{x})}|\vec{\alpha}_{\mathtt{x}})^2} \right) \right\} + \frac{k}{2} \quad \text{bits} \tag{5.4}$$

Recall the Fisher information of $\vec{\Theta}_{n(\mathtt{x})}$ by Equation 3.21. Also Equation 5.1 provides the likelihood of $\vec{\alpha}_{\mathtt{x}}$ that describes $\vec{\Theta}_{n(\mathtt{x})}$. Accordingly,

$$det[Fisher(\vec{\Theta}_{n(\mathtt{x})})] = \frac{X_{n(\mathtt{x})}^{k-1}}{\left(\prod_{i=1}^{k-1} \theta_i \right) \left(1 - \sum_{i=1}^{k-1} \theta_i \right)}$$

where $X_{n(\mathtt{x})}$ is the total number of state transitions from state $\mathtt{x}$ to any other state ($\mathtt{x} \rightarrow *$) found in an alignment $\mathcal{A}_n$, and $\theta_i \in \vec{\Theta}_{n(\mathtt{x})}$ is the MML estimate for the probability of a state transition $\mathtt{x} \rightarrow i$. Note: state $\mathtt{m}$ has $X_{n(\mathtt{m})}$ as the total number of $\mathtt{m} \rightarrow \mathtt{m}$, $\mathtt{m} \rightarrow \mathtt{i}$ and $\mathtt{m} \rightarrow \mathtt{d}$ transitions, while the denominator refers to the product of MML estimates for Pr($\mathtt{m}|\mathtt{m}$) and (1-Pr($\mathtt{m}|\mathtt{m}$)) (i.e. Pr($*|\mathtt{m}$))

over $\mathcal{A}_n$. Similarly, for state $\mathtt{i}$, $X_{n(\mathtt{i})}$ is the total number of $\mathtt{i}\rightarrow\mathtt{i}$, $\mathtt{i}\rightarrow\mathtt{m}$, $\mathtt{i}\rightarrow\mathtt{d}$, $\mathtt{d}\rightarrow\mathtt{d}$, $\mathtt{d}\rightarrow\mathtt{m}$ and $\mathtt{d}\rightarrow\mathtt{i}$ transitions, with the denominator referring to the product of MML estimates for Pr($\mathtt{i}|\mathtt{i}$), Pr($\mathtt{m}|\mathtt{i}$) and (1-Pr($\mathtt{i}|\mathtt{i}$)- Pr($\mathtt{m}|\mathtt{i}$)) (i.e Pr($\mathtt{d}|\mathtt{i}$)) over $\mathcal{A}_n$. Each $\theta_i \in \vec{\Theta}_{n(\mathtt{x})}$ is computed using the Equation 3.25 derived in §3.3.3 for MML based multi-state model estimation over a Dirichlet prior. Accordingly,

$$\theta_i = \left(\frac{x_{n(\mathtt{x}),i} + \alpha_{\mathtt{x},i} - \frac{1}{2}}{X_{n(\mathtt{x})} + \kappa_{\mathtt{x}} - \frac{k}{2}} \right)$$

where, $x_{n(\mathtt{x}),i}$ is the number of state transitions $\mathtt{x}\rightarrow i$, and $\alpha_{\mathtt{x},i}$ is the corresponding component (i.e. pseudocount) for a state i in the Dirichlet prior parameter vector $\vec{\alpha}_{\mathtt{x}}$. Formally, we can also denote the full parameter vector (with free parameters and dependent parameter together) by $\bar{\theta}_i \in \{\vec{\Theta}_{n(\mathtt{x})}, 1 - \sum_{i=1}^{k-1} \theta_i\}$.

The second part $I(\mathbf{A}_{(\mathtt{x})}|\vec{\Theta}_{(\mathtt{x})}, \vec{\alpha}_{\mathtt{x}})$ deals with transmitting 3-state data $\mathbf{A}_{(\mathtt{x})}$ of each alignment $\mathcal{A}_n \in \mathbf{A}$ relevant to state $\mathtt{x}$, using the parameters $\vec{\Theta}_{n(\mathtt{x})}$ and $\vec{\alpha}_{\mathtt{x}}$ stated in the first part of the message. Accordingly:

$$I(\mathbf{A}_{(\mathtt{x})}|\vec{\Theta}_{(\mathtt{x})}, \vec{\alpha}_{\mathtt{x}}) = \sum_{n=1}^{N} \left\{ I(\mathcal{A}_n|\vec{\Theta}_{n(\mathtt{x})}, \vec{\alpha}_{\mathtt{x}}) + \left(\frac{k-1}{2} \right) \right\} \quad \text{bits} \tag{5.5}$$

with $I(\mathcal{A}_n|\vec{\Theta}_{n(\mathtt{x})}, \vec{\alpha}_{\mathtt{x}}) = \sum_{i=1}^{k} \left(x_{n(\mathtt{x}),i} \times -\log_2 \left(\bar{\theta}_i \right) \right)$, where $\bar{\theta}_i \in \{\vec{\Theta}_{n(\mathtt{x})}, 1 - \sum_{i=1}^{k-1} \theta_i\}$

The following sections present the specifics related to a one-time exercise of estimating the optimal three-state machine as a function of evolutionary time, using Dirichlet probability distributions and their estimation introduced in the above sections.

5.3 Estimation of Time-dependent Three-state Machines

The inference of alignment three-state machine as a function of evolutionary distance depends on a randomly sampled, reliable structural alignment dataset. All those alignments were grouped by their optimal evolutionary time $t \in [1, 1000]$ using the PAM (Dayhoff et al., 1978) (Φ_{PAM}) series of substitution matrices). Each $\mathbf{M}_t \in \Phi_{\text{PAM}}$ provides transition probabilities that suitably explain amino acid substitutions occurring during some discrete evolutionary time t. (Details of how these matrices behave as a function of t are comprehensively discussed in Chapter 6. The search for the optimal t of a protein sequence pair was done using the iterative quaternary search method explained in §4.1.2, over all matched amino acid pairs. Next, the MML estimation method described in §5.2.2 was used to obtain optimal 1-simplex and 2-simplex Dirichlet models for each group of alignments. This allowed a t versus $\mathbf{\Theta}$ mapping to be included as part of the *codebook* defined in the MML protein alignment framework introduced in Chapter 4.

The following section elaborates on the random sampling and filtering procedure employed for preparing a reliable benchmark alignment dataset prior to Dirichlet model estimation.

5.3.1 Choosing a Source Collection of Protein Structural Domain-pairs

The quality and reliability of time-parameterised Dirichlet priors solely depend on: (1) the source collection of protein alignments, and (2) the quality of their specified sequence relationships/alignments. Thus, a main requirement in selecting the benchmark alignment dataset was to include protein domain pairs with a detectable amino acid relationship and a sufficient representation of varying sequence distances (evolutionary times).

Initially, a set of 118,384 unique protein domain pairs were randomly sampled from the Structural Classification of Proteins (SCOP - version 2.07) database (Murzin et al., 1995) (discussed in §2.2.3). Since our interest is on proteins with related amino acid sequences, the random selection of those pairs was restricted to the SCOP levels of `superfamily` and `family`. The domains within the same `family` are often closely related in their sequence, while those from the same `superfamily` but under different `families` contain sequences which have diverged, yet carrying (often) a detectable sequence relationship. Accordingly, the source collection of protein domain pairs is composed of 47,687 pairs that are related at the `family` level, and 70,697 pairs that are related at the `superfamily` level.

Below describes the random sampling method used to arrive at this inceptive data source for the task at hand.

Random sampling method:

A domain pair is randomly selected by utilising the SCOP organisation of domains within its hierarchical classification tree. The internal nodes of this tree are associated with the four-level classification of protein domains (specified in §2.2.3). Each domain is uniquely represented by a leaf node. A traversal from the root node to a leaf node yields a domain.

The sampling procedure involves traversing from the root to a leaf node, passing each of the SCOP levels: `class`, `fold`, `superfamily` and `family` in order. At each node of this traversal, until a leaf node (domain) is reached, a child node is selected from the available children (i.e. nodes in the level below the current node), by sampling randomly based on the weights (i.e. number of leaves) of their respective subtrees.

Thus, to identify domain pairs within the same `superfamily` but under different families, the traversal first proceeds from the root until it reaches the level of `superfamily`. Then the weighted random sampling method selects two random domains (leaves) from two different families (child nodes), while considering only the superfamilies with ≥ 2 families. Similarly, to identify pairs from the same `family`, when the traversal reaches `family` level nodes, a pair of its children (leaves) are randomly selected, while only considering families with ≥ 2 domains.

Preparing alignments for the source collection:

Since all domain pairs chosen from the SCOP database have their associated three-dimensional atomic coordinate information, the pairs can be structurally aligned to decipher amino acid residue-residue correspondences. The reliability of these correspondences are more trustworthy. This is because, functional constraints on evolving protein domains ensure that their structures are far more conserved than their amino acid sequences (Lesk, 2016) (See §2.2.1 for discussion). Thus, each sampled SCOP domain pair was structurally aligned using the MMLigner structural alignment program (Collier, 2016). This generated an initial benchmark dataset of 118,384 three-state alignment strings.

Further filtering and preprocessing steps:

The above prepared structural alignment collection was filtered and preprocessed for later modelling exercises (discussed in Chapter 6), to arrive at a more reliable benchmark dataset. First, sequence pairs with an optimal $t = 0$ evolutionary time (i.e. accounting for 100% sequence identity) were removed, bringing down the dataset size to 111,009. Next, the filtering criteria only retained alignments that have a Root-mean-square deviation (RMSD) ≤ 5, with at least 60 residues of coverage (i.e. the number of matched pairs) and no chain breaks. The special amino acid symbols (i.e. ones other than the naturally occurring 20 amino acid symbols – '$','B','O','U','X','Z','/') were ignored as well. Further they were preprocessed to exclude terminal gaps.

The above preprocessing exercise resulted in a final dataset of 59,092 structurally aligned protein domain pairs, with 22,720 at `family` level and 36,372 at `superfamily` level. The average sequence length is 188.352, and the average sequence identity percentage is 26.447%. This covers 96,129 unique SCOP domains across all pairs, with their alignments containing a total number of 8,145,678 substitution pairs and 3,327,218 indels. 17.89% of these pairs cover same-species relationships, while the entire set covers 2801 different species. The main representative is *Homo sapiens* (9.14%), followed by *Saccharomyces cerevisiae* (1.74%), *Escherichia coli* (1.35%), *Mus musculus* (1%), *Bos taurus* (0.2%),*Staphylococcus aureus* (0.2%) and others. Out of the available kingdom information for 18,287 pairs, 15,426 were three types of same kingdom comparisons, with the main representation being *Metazoa-Metazoa* (13,458 pairs) followed by *Fungi-Fungi* (1,354), and *Viridiplantae-Viridiplantae* (614). Out of 47,217 pairs that had phylum information, same *phylum-phylum* comparisons have the primary representation of *Chordata* (12,020 pairs), followed by *Proteobacteria* (4,658 pairs) and *Ascomycota* (1,235 pairs). All kingdom, phylum and species information were retrieved from UniProt.

5.3.2 Searching for the Optimal Dirichlet Parameters

Given a benchmark alignment dataset of SCOP domain pairs (prepared as explained in the previous section), the next step of estimating the time-parameterised three-state machine is to search for the optimal Dirichlet models that explain the three-state strings in this dataset.

Recalling §4.1.1 with the finite state machine illustrated in Figure 4.1, the free parameter $\Pr(\mathtt{m}|\mathtt{m})$ related to the `m` state, and $\{\Pr(\mathtt{i}|\mathtt{i}),\Pr(\mathtt{m}|\mathtt{i})\}$ free parameters related to `i` state (equivalently `d` state), lie in a unit 1-simplex and unit 2-simplex, respectively. Accordingly, estimating the $\Pr(\mathtt{m}|\mathtt{m})$ parameter involves the inference of a Dirichlet model describing a unit 1-simplex. Similarly, estimating $\Pr(\mathtt{i}\,|\mathtt{i})$ and $\Pr(\mathtt{m}\,|\mathtt{i})$ parameters involve the inference of a Dirichlet model that describes a unit 2-simplex.

Hence, our aim is to estimate time-parameterised 1-simplex and 2-simplex Dirichlet models, by searching for their optimal parameters $\vec{\alpha}_{\mathtt{m}}$ and $\vec{\alpha}_{\mathtt{i}}$, for each group of alignments that represents a distinct evolutionary time $t \in [1, 1000]$ under the PAM model of evolution. Optimising the parameters of 1-simplex and 2-simplex Dirichlet models are carried out independently under the general MML objective function given by the Equation 5.3. This thesis implements two possible approaches to minimise the objective function: (1) *an exhaustive method* (to some fixed precision of parameters), and (2) *a Markov Chain Monte Carlo (MCMC) method.*[7] The exhaustive approach is used in this Chapter as the initial inference method, which is then replaced later at Chapter 6 by the MCMC method.

[7]Note: There also exists maximum likelihood based approaches for Dirichlet model estimation. For instance, Ye et al. (2011) presented an information theory based method using the minimum description length principle. This thesis formulates the problem under the minimum message length criterion.

5.3.3 Exhaustive Method

This method refers to sweeping a viable subspace of the Dirichlet parameter space of $\vec{\alpha}$ exhaustively, under a fixed parameter precision. The reparameterisation given in Equation 5.2.1 enables this to be processed in terms of the mean vector $\vec{\mu}$ and concentration κ. The iterative search goes as follows.

When estimating the 1-simplex Dirichlet model for the `match` state ($\Pr(\texttt{m}|\texttt{m})$), the search sweeps over the range $0.7 \leq \mu_1 < 1.0$ with 0.001 increments. (A lower limit of $\mu_1 = 0.7$ is a fair choice since it associates a sensible average length of 3 for a match block). For each possible value of μ_1, κ is explored in the range $(\kappa_{lo}, \kappa_{hi}]$, where κ_{lo} is a lower bound to ensure that all components of $\vec{\alpha} > 1$ (this in turn ensures the availability of a mode vector). κ_{hi} is an upper bound that limits Dirichlet model from acquiring a highly concentrated, sharp peak. κ_{lo} is chosen such that, if $(\frac{1}{\mu_1} < \frac{1}{\mu_2})$, $\kappa_{lo} = \lceil \frac{1}{\mu_2} \rceil$, otherwise, $\kappa_{lo} = \lceil \frac{1}{\mu_1} \rceil$. κ_{hi} was chosen to be 600.

When estimating the 2-simplex Dirichlet model for the `insert` state ($\Pr(\texttt{i}\ |\texttt{i})$ and $\Pr(\texttt{m}\ |\texttt{i})$), the parameter space is explored for each $\kappa \in (3, 100)$, by setting lower bounds for μ_1 and μ_2 as $\frac{1}{\kappa}$. The increments happen by 0.01.

5.3.4 Markov Chain Monte Carlo Method

This approach refers to the Monte Carlo method of *Metropolis-Hastings* algorithm and Simulated Annealing (SA) (Metropolis et al., 1953; Kirkpatrick et al., 1983) for Dirichlet parameter estimation. It is also used in Chapter 6 for a separate purpose.[8] This section first gives a preliminary view to the general SA process, and then go into describing how it was employed for finding optimal Dirichlet parameters.

Simulated Annealing

Simulated Annealing (Kirkpatrick et al., 1983) is a heuristic search based on an analogy of cooling solids (into a minimum energy state). The system starts with a random arrangement (i.e. state) at a high temperature, and a cooling schedule is run to gradually decrease the temperature. As the system cools down, the molecules rearrange as to obtain a more stable, lower energy state. For a fixed temperature T at thermal equilibrium, the energy distribution of all possible states are described by a Boltzmann distribution. Thus, the probability of the system in a certain energy state E is:

$$\Pr(E) \propto e^{-\frac{E}{RT}} \quad \text{where } R = \text{Boltzmann constant}$$

As the temperature decreases, the amount of molecules in higher energy states also drops, realising lower energy states to be more probable for the system, shifting the distribution to be more left skewed.

Given a temperature T and a random system state as the initial state, a *Metropolis Monte Carlo method* (Metropolis et al., 1953) simulates the evolution of the system state into a thermal equilibrium. The process perturbs the current system state to generate a new state. (This perturbation happens such that the next state is a neighbouring state – not too different from the current state). If the new state is lower in energy than the current state (i.e. if the energy difference $\Delta E > 0$), it is accepted and the perturbation process continues with the new state. Otherwise it is accepted with a probability $e^{-\frac{\Delta E}{RT}}$, referred to as the *Metropolis criterion*. After many iterations, the probability distribution approaches the Boltzmann distribution. Next, the temperature is slightly lowered and the Monte Carlo method is repeated until a new thermal

[8] For deriving optimal Markov amino acid substitution models in §6.3

equilibrium with a new Boltzmann distribution is reached. This process is iterated until some minimum threshold temperature is reached. At higher temperatures, more accepts can be expected from the Metropolis criterion.

The Monte Carlo method of evolving the system towards thermal equilibrium is analogous to an irreducible and reversible Markov chain of energy states, reaching its stationary distribution (See §6.2 for more details on Markov models and their properties). This Markov model satisfies the following equation (known as the detailed balance):

$$\begin{aligned} &\Pr(x)\Pr(x \to y \text{ generation})\Pr(x \to y \text{ acceptance}) \\ &= \Pr(y)\Pr(y \to x \text{ generation})\Pr(y \to x \text{ acceptance}) \end{aligned}$$

Commonly, a microscopic reversibility is imposed by a symmetric probability for a trial move, where $\Pr(x \to y \text{ generation}) = \Pr(y \to x \text{ generation})$(Frenkel et al., 2017). Consequently, the detailed balance equation becomes:

$$\frac{\Pr_{\text{accept}}(x \to y)}{\Pr_{\text{accept}}(y \to x)} = \frac{\Pr(y)}{\Pr(x)}$$

The Metropolis acceptance rule is the most common amongst many others that satisfy the above criterion (Frenkel et al., 2017):

$$\underset{\text{accept}}{\Pr}(x \to y) = \min\left\{1, \frac{\Pr(y)}{\Pr(x)}\right\}$$

Accordingly, the acceptance of the new state is always guaranteed with a probability of 1, when the new state is more probable than the current state. In contrast, if the new state is less probable than the current state, the ratio of the stationary probabilities are taken such that, if $\Pr(y)$ is closer to $\Pr(x)$, it is more likely to be accepted.

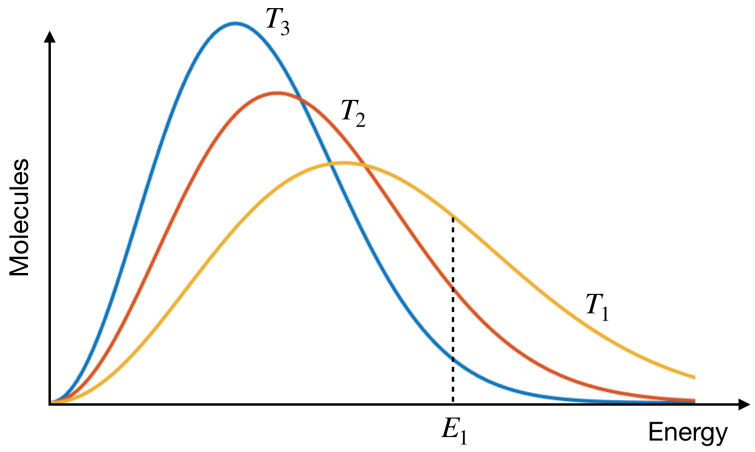


Figure 5.3: An example plot describing how state energy distribution (Maxwell-Boltzmann distribution) at thermal equilibrium moves as the temperature T decreases ($T_3 < T_2 < T1$). The same energy state E_1 becomes less probable when T drops. Analogous to this cooling process, simulated annealing optimisation explores higher energy states at high T, with the aim of overcoming local optima and moving into different regions in the solution landscape. (Note: The curves in this plot were generated using the equations defined at http://web.mit.edu/8.13/matlab/Examples/maxboltz.m for illustration purpose)

"Monte Carlo method is a stochastic trajectory through configuration space" (Attard, 2002). The objective is to generate a sequence of states from the system state space. The naive method (i.e. a random walk around the states) treats all states as equiprobable. However, a selection bias based on the energy of the state is important such that the lower energy states are preferred.

In SA, where we consider the state energy distribution at thermal equilibrium to be Boltzmann, the ultimate goal of MCMC under a given temperature T is to reach this distribution as the stationary distribution, which reflects the lowest energy state under the defined T. Therefore, for any state x_i for which the energy of the system is E_i, we have:

$$\Pr(x_i) \propto \exp\left\{\frac{-E_i}{RT}\right\}$$

This gives:

$$\Pr_{\text{accept}}(x \to y) = \min\left\{1, \exp\left(\frac{-\Delta E}{RT}\right)\right\}$$
$$\text{where } \Delta E = E_y - E_x$$

When T decreases, the stationary distribution changes, with lower energy states becoming more probable than before. Attard (2002) states that: *"The Metropolis algorithm has the effect of keeping the trajectory that the system follows through configuration space close to the energy valley floor; at lower temperatures, lower energy states dominate; at higher temperatures, more energetic configuration become increasingly accessible"*. As the temperature goes down, the system becomes more stable with identifying lower energy states, and therefore, the number of accepts can be expected to reduce. If it is an energy increase, the state is accepted with a probability proportional to the Boltzmann factor of the change in energy (i.e. ratio of the current state probability to new state probability).

This thesis applies the above described Simulated Annealing method at two instances. The following section relates to the first instance, that is to find the optimal Dirichlet parameters which minimise the objective function defined by Equation 5.3. The second instance is discussed under Chapter 6 which utilises it for optimising Markov models of amino acid substitution.

MCMC Search for the Optimal Dirichlet parameters

The search for the optimal Dirichlet parameters followed the Simulated Annealing approach described above, by running a single Markov chain of evolution until it reaches its stationary distribution under a temperature value of 1.[9] It is analogous to evolving a system towards thermal equilibrium under a constant temperature, starting from some initial state. A state reflects the energy of the system. Equivalently, in the context of Dirichlet estimation, a state of the system refers to some $\vec{\alpha}_{\mathtt{x}}$ parameter vector of the Dirichlet model related to the state $\mathtt{x} \in \{\mathtt{m}, \mathtt{i}\}$. The energy of the state is explained by our objective function (Equation 5.3). The

[9]This MCMC search is used in Chapter 6 to infer Dirichlet distributions under several different substitution models. It begins with a reasonable starting point for Dirichlet parameters, based on the ones obtained in this Chapter using the exhaustive method under the PAM substitution model. These parameters are in the near-neighbourhood to those that can be inferred using varying substitution models. Thus, the inference can be made efficient by not running the entire cooling schedule of the simulated annealing process, but a single Markov chain of evolution under a constant temperature of 1.

Function `Perturb_Dirichlet_Param`($\vec{\alpha}_x$):
  $\{\kappa,\vec{\mu}\} \leftarrow$ `decompose`($\vec{\alpha}_{\mathtt{x}}$);
  $u_1 \leftarrow$ `random_uniform`$(0,1)$;
  if $u_1 \leq 0.5$ **then**
    $\hat{\mu} \leftarrow$ `Perturb_Mean`($\vec{\mu}$,$\langle$**state**$\rangle$);
  end
  else
    $\kappa \leftarrow$ `Perturb_Concentration`(κ);
  end
return $\kappa.\vec{\mu}$

Function `Perturb_Mean`($\vec{\mu}$,$\langle$**state**$\rangle$):
  if $\langle$**state**$\rangle$ = match **then**
    $\bar{\kappa} \leftarrow 10000$;
  end
  else
    $\bar{\kappa} \leftarrow 1000$;
  end
return `sample_dirichlet`($\bar{\kappa}.\vec{\mu}$)

Function `Perturb_Concentration`(κ):
  $\delta \leftarrow$ `random_uniform`$(0.1, 10)$;
  $u_2 \leftarrow$ `random_uniform`$(0,1)$;
  if $u_2 \leq 0.5$ **then**
    $\kappa \leftarrow \kappa + \delta$;
  end
  else
    $\kappa \leftarrow \kappa$ - δ;
  end
return κ

Function `sample_dirichlet`($\vec{\alpha}$):
  Init $\vec{y} \leftarrow \vec{0}$, sum $\leftarrow 0$;
  for $(i = 1 \rightarrow |\vec{\alpha}|)$ **do**
    $y_i \leftarrow$ `gamma_random`$(\vec{\alpha_i}, 1)$;
    sum $\leftarrow$ sum $+ y_i$;
  end
  for $(i = 1 \rightarrow |\vec{\alpha}|)$ **do**
    $y_i \leftarrow y_i/$sum ;
  end
return $\vec{y}$

Figure 5.4: Pseudocode for perturbing the parameter of a Dirichlet distribution

Markov chain is evolved by perturbing the current state. The perturbed state is evaluated for acceptance as the next state. If it is better in energy (i.e. the total message length is lesser than that of the current state), the perturbed state is accepted as the next state in the chain, with a probability of 1. If not, it is accepted with a probability of $e^{\Delta E}$, where ΔE is the energy difference between the current state and the perturbed state (*Metropolis criterion*). This iterative process is continued for a reasonable number of iterations. If no more improvements are made (i.e when the rejection rate is very high), it is an indication that the chain is closer to its equilibrium state.

The Dirichlet parameter $\vec{\alpha}_{\mathtt{x}}$ is perturbed as per the `Perturb_Dirichlet_Param()` function defined in the pseudocode presented in Figure 5.4, utilising the reparameterisation given in Equation 5.2.1. The `Perturb_Mean()` function perturbs the mean vector of Dir($\vec{\alpha}_{\mathtt{x}}$), $\vec{\mu}$, by sampling a new probability vector under a Dirichlet distribution with mean $\vec{\mu}$ and some concentration $\bar{\kappa}$. `sample_dirichlet()` function depicts the sampling approach stated in §5.2.1. Here we choose $\bar{\kappa} = 10,000$, a higher concentration for state `m`, compared to $\bar{\kappa} = 1000$ for state `i`. This is because, the 1-simplex model has less freedom due to just one free parameter, making it more sensitive to a slight perturbation. On the other hand, 2-simplex model has two degrees of freedom, thus we allow more flexible perturbations. As the Dirichlet concentration decreases, the neighbourhood where a perturbed sample resides in the distribution expands. See Table 5.1 for the average root-mean-square deviation (RMSD) between an example original vector and its perturbed vector over different concentration values. Note: RMSD between two vectors, $\vec{x}$ and $\vec{y}$ of size k is: $\sqrt{\frac{1}{k}\sum_{i=1}^{k}(x_i - y_i)^2}$.

For instance, let $\vec{\mu} = \{0.8, 0.2\}$ for state `x=m`. If we perturb this vector $10,000$ times by defining Dir$(10000.\vec{\mu})$ as per the above function, it results in a new probability vector

with a mean squared distance of 0.003190. On the other hand, the same experiment with $\vec{\mu} = \{0.3, 0.2, 0.5\}$ for state `x=i` under Dir$(1000.\vec{\mu})$ gives a 0.012654 mean squared distance.

Table 5.1: The average RMSD between the mean vector $\vec{\mu}$ and a sampled probability vector from Dir$(\vec{\alpha})$ with $\vec{\alpha} = \bar{\kappa} \cdot \vec{\mu}$, over 10,000 random samples, for two example mean vectors representing state `x=m` and state `x=i`

$\bar{\kappa}$	$\vec{\mu} = \{0.8, 0.2\}$ (`x=m`)	$\vec{\mu} = \{0.3, 0.2, 0.5\}$ (`x=i`)
10	0.097035	0.122278
100	0.031632	0.039859
1000	0.010089	0.012654
10,000	0.003190	0.004000
100,000	0.001008	0.001265

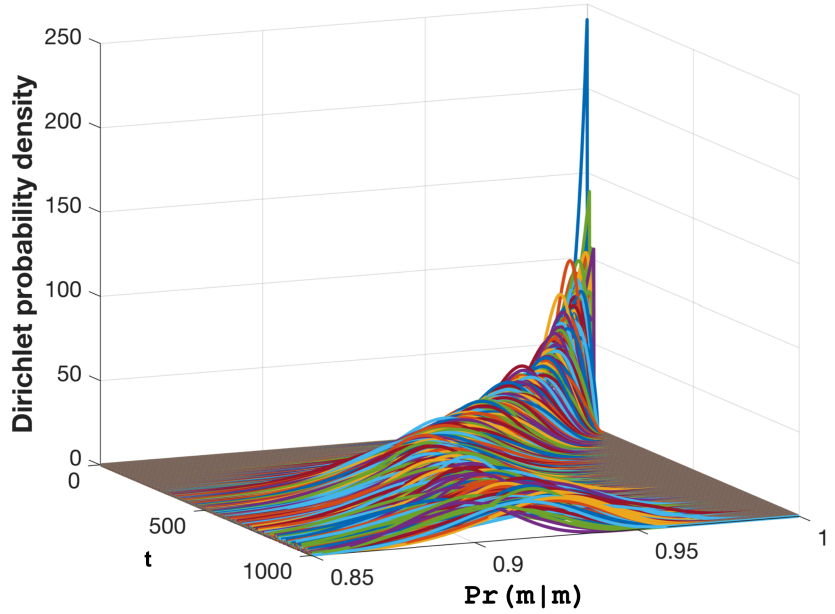
5.3.5 Results and Insights

Initially, the *exhaustive approach* described in §5.3.3 was taken over the inceptive dataset of 118,384 structural alignments to arrive at optimal Dirichlet distributions for each $t \in [1, 1000]$ under the PAM series of substitution matrices.

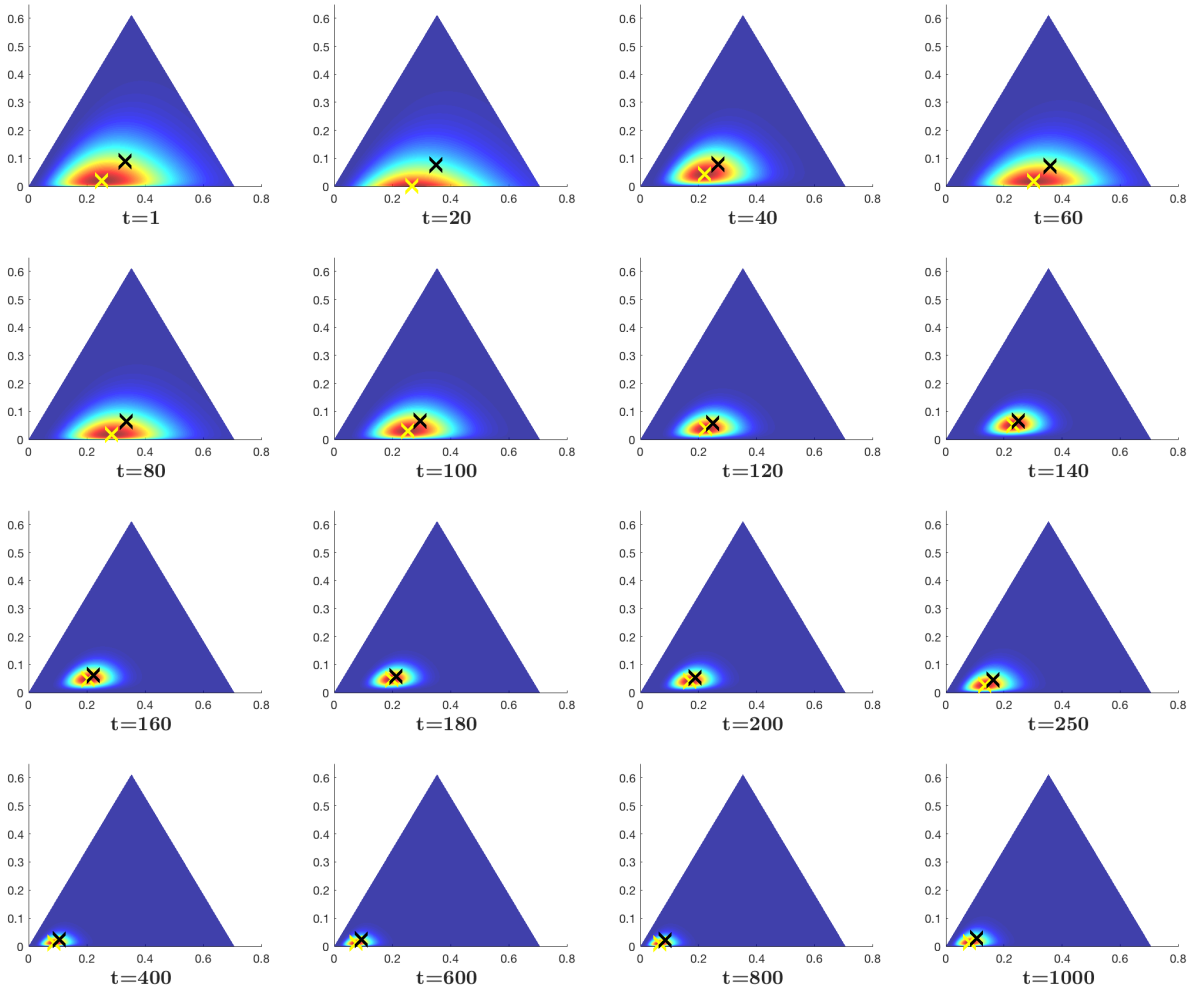
Figure 5.5 illustrates how these models vary as a function of t, clearly depicting a correlation. As shown in Figure 5.5a, Pr(`m`|`m`) follows a decreasing trend, starting from a very sharp probability value at almost 1 for $t = 1$, gradually flattening the distribution while shifting away towards lower probability values.

In contrast, Figure 5.5b shows a skewed shift in the distribution towards the left corner of the unit 2-simplex, while becoming more concentrated as time t increases. Only sixteen 2-simplex models covering the range $t \in [1, 1000]$ are visualised out of all thousand models. The heat map shows the inferred concentration of probability density about the mode vector (marked by a yellow colour cross). The black cross shows the mean under the same distribution. A triangle visualisation denotes the 2-simplex support for the $\mathbb{L}1$-normalised transition probability vectors of the `insertion` state. Its three corners (bottom-left, bottom-right, top) show points where Pr(`i`|`i`), Pr(`m`|`i`) and Pr(deletion|`i`) individually becomes exactly 1, while remaining two become 0. Figure 5.5b has truncated the unit-simplex support to clearly see the differences (thus, only the bottom-left corner is visible). Altogether, the plots indicate an increase in Pr(`i`|`i`) as the evolutionary time t increases (see how the mean and mode vectors approach the bottom-left corner).

Insights on the expected matched block and gap lengths: The Pr(`m`|`m`) parameter influences the observed lengths of the matched blocks produced by the three-state machine. These lengths are geometrically distributed, and the probability of seeing a matched block of length L using this parameter is: $(1 - \Pr(\texttt{m}|\texttt{m})) \times \Pr(\texttt{m}|\texttt{m})^{L-1}$. The expected length of a matched block is given by $\frac{1}{1-\Pr(\texttt{m}|\texttt{m})}$. Furthermore, with the enforced symmetry of state transitions from a `match` state to an `insertion` state or `deletion` state, we have $1 - \Pr(\texttt{m}|\texttt{m}) = 2\times \Pr(\texttt{i}|\texttt{m}) = 2\times \Pr(\texttt{d}|\texttt{m})$. This value informs the probability of observing the start of a gap at a given position in an alignment produced by the state machine, informing the frequency of gaps to expect. Figure 5.6a plots the expected probability of observing a gap as a function of time t, when Pr(`m`|`m`) is set to the mean and mode values under the respective Dirichlet distributions. The plot evinces an approximately linear increase in probability of a gap open, within t in the range $[1, 350]$. This linear trend agrees with the observations by Gonnet et al. (1992); Benner et al. (1993). Afterwards it acquires a rather stable value.

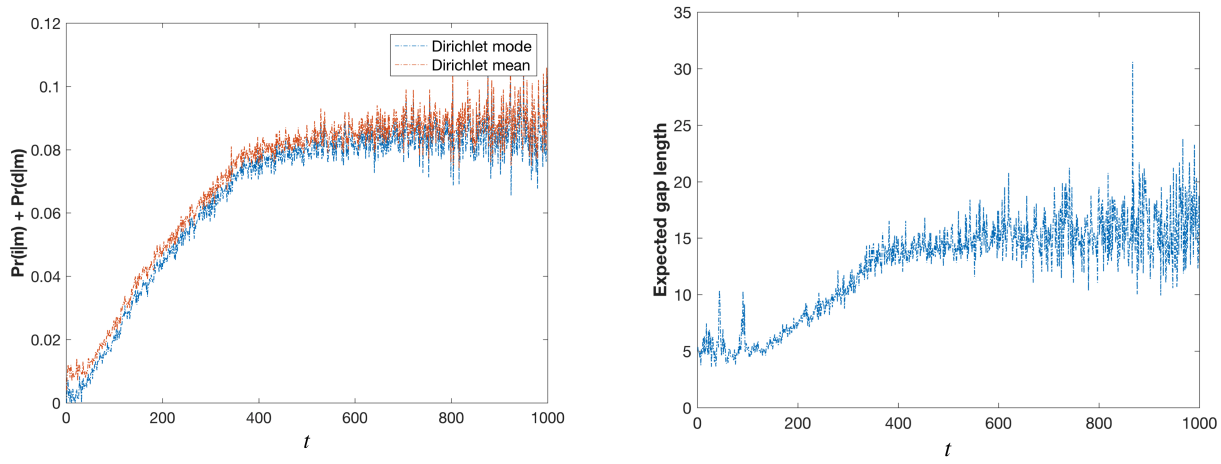


(a) 1-simplex Beta plots



(b) 2-simplex Dirichlet plots

Figure 5.5: Visualisation of the inferred Dirichlet distributions, modelling the three free parameters of the finite state machine illustrated in Figure 2.4b. (a) 1-simplex distributions of Pr(m|m) associated with state m, as a function of evolutionary time $t \in [1, 1000]$. (b) 2-simplex distributions of Pr(i|i) and Pr(m|i) associated with state i (and by symmetry, state d), as a function of evolutionary time $t = \{1, 20, 40, 60, 80, 100, 120, 140, 160, 180, 200, 250, 400, 600, 800, 1000\}$ under the PAM Markov model of amino acid substitutions. The yellow cross marker and black cross marker highlight the mode and mean vectors, respectively.



(a) (b)

Figure 5.6: (a) The distribution of mean (red) and mode (blue) values of $1-\Pr(\mathtt{m}|\mathtt{m})$ under the 1-simplex Dirichlet priors, as a function of evolutionary time t under the PAM substitution model. (b) The distribution of expected gap lengths derived from the mode estimate of $\Pr(\mathtt{i}|\mathtt{i})$ under the inferred 2-simplex Dirichlet priors.

On the other hand, the $\Pr(\mathtt{i}|\mathtt{i})$ parameter controls the length of a gap (i.e. block/stretch length of insertions or, symmetrically, deletions). Again, these block lengths are geometrically distributed with the probability of observing a gap of length L is defined by $1 - \Pr(\mathtt{i}|\mathtt{i}) \times \Pr(\mathtt{i}|\mathtt{i})^{L-1}$. Their expected gap length is given by: $\frac{1}{(1-\Pr(\mathtt{i}|\mathtt{i}))}$. Figure 5.6b plots this as a function of t with mode estimates. On average, the expected gap length is about 5 amino acid residues for t in the range $[1, 120]$, and, barring a few outliers at $t = [43, 44, 45, 91, 94]$, the trend is flat. Examining the structural alignments of these outliers in the dataset, we find protein domain pairs with circularly permuted amino acid sequences and other pairs with plastic deformations in their structures. Note that a circular permutation between proteins results in a non-sequential relationship. Enforcing a sequence alignment on such a relationship yields regions that cannot be sequentially-aligned, which are then misinterpreted as long gaps. Similarly, domain pairs with plastic deformations have the same effect on gap lengths.

Further, in the range of $t \in [120, 350]$, we observe that the expected gap length increases linearly, as a function of t. Subsequently, for $t > 350$, the gap length again shows a flat trend line, averaging on about 13 to 15 residues. The linear trend contradicts the observations of Benner et al. (1993) that the expected gap length decreases with an increase in PAM time.

In summary, with these evolutionary-time-parameterised Dirichlet models, we are now able to improve the MML protein alignment framework introduced in Chapter 1, for generating more realistic sequence alignments. As described in §4.1.1, these models can either be used as

- priors for the MML87 method (Equation 3.25) of estimating the state machine parameters $\vec{\Theta}$ for a given alignment string (**Approach 1**), or,
- to obtain their expected values (e.g. mean vector or mode vector) for their direct usage as $\vec{\Theta}$ parameters (**Approach 2**)

Approach 1 requires the statement of $\vec{\Theta}$ within the message itself (Equation 4.5), yielding $I(\vec{\Theta})$ bits for the first part of the message in Equation 4.1. On the other hand, **Approach 2** is more efficient since the time-parameterised Dirichlet models are part of the *codebook* and so are their expected values. Thus, it is chosen for the subsequent experiments.

5.4 Evaluation of the Improved Alignment Framework over Benchmark Datasets

This section discusses an experiment conducted to evaluate the performance of the improved MML protein alignment framework under **Approach 2**. Here, the mode vectors of the above time-parameterised Dirichlet models were used to describe the state transition probabilities of the three-state machine.

Two types of benchmark datasets were employed to test the performance of the MML protein alignment framework under the derived mapping between the optimal PAM-t evolutionary time and state machine parameters. The first is a set of experimentally verified remote orthologs reported by Szklarczyk et al. (2012), containing a total of 877 protein sequence pairs spread over two groups. The second is the entire *twilight zone* dataset from SABmark (Van Walle et al., 2005) containing 10,250 protein sequence pairs spread over 209 groups. Results are presented in Table 5.2 and Table 5.3.

Specifically, the dataset of Szklarczyk et al. (2012) includes, in the first group, 405 pairs of human orthologous mitochondrial proteins found in *Saccharomyces cerevisiae*, and, in the second group, 472 pairs of orthologs found in *Schizosaccharomyces pombe*. Their average percentage sequence identity is reported as 27.7%, with about 40% of this dataset having pairs of $< 25\%$ sequence identity, and 36% having pairs of 25-35% sequence identity.

The MML framework was tested in terms of both the *optimal alignment model* and the *marginal probability model*.

The optimal model ($\text{MML}_{\text{Optimal}}$) finds the best alignment hypothesis that minimises the total message length given by the Equation 4.1, yielding the $I(\mathcal{A}^*, \langle \mathbf{S}, \mathbf{T} \rangle)$ statistic for each pair of proteins in the dataset. This is together with the information measure of the alignment complexity ($I(\mathcal{A}^*)$), and the information measure of fidelity of the alignment to explain the corresponding sequence data using its specified relationship ($I(\langle \mathbf{S},\mathbf{T} \rangle | \mathcal{A}^*)$.

The *marginal probability model* ($\text{MML}_{\text{Marginal}}$) estimates the negative logarithm of the marginal probability as per the Equation 4.10, that gives the $I_{\text{Marginal}}(\langle \mathbf{S}, \mathbf{T} \rangle)$ statistic. Both $I(\mathcal{A}^*, \langle \mathbf{S},\mathbf{T} \rangle)$ and $I_{\text{Marginal}}(\langle \mathbf{S},\mathbf{T} \rangle)$ are compared with their corresponding $I_{\text{NULL}}(\langle \mathbf{S},\mathbf{T} \rangle)$ *null model* message length, yielding the bits of 'compression' statistic.

Further, the results of these MML alignment models were compared against seven widely-used protein sequence alignment programs: ClustalW (Larkin, 2007), CONTRAlign (Do et al., 2006), KAlign (Lassmann and Sonnhammer, 2005), MAFFT (Katoh and Standley, 2013), MUSCLE (Edgar, 2004), ProbCons (Do et al., 2005), and T-COFFEE (Notredame et al., 2000). The performance across all these programs was evaluated using the 'Compression' statistic against the *null model*. For each alignment produced by one of the programs, its $I(\mathcal{A}^*, \langle \mathbf{S},\mathbf{T} \rangle)$ value was computed by finding the best parameters that minimises the value. We consider as 'hits' (i.e. correctly identified as related) only those alignments that give a positive compression. This allows us to compute the percentage of the total number of sequence pairs that passes the null hypothesis test for significance (%-Hits).

Table 5.2 presents the results across the benchmark of human remote orthologs (Szklarczyk et al., 2012). It reports the corresponding median values (across the entire group) against $I(\mathcal{A}^*)$, $I(\langle \mathbf{S},\mathbf{T} \rangle | \mathcal{A}^*)$, and 'Compression' entries. The $\text{MML}_{\text{Marginal}}$ model has the highest percentage of hits, (94.57% and 94.49% across the two groups, respectively). This is followed by $\text{MML}_{\text{Optimal}}$ to identify the best alignment hypothesis (79.26% and 80.51%). The next best couple are MUSCLE and KAlign (both 73.83% hits in the first group; 76.06% and 74.79% hits, respectively in the second group).

Table 5.2: Comparison of various alignment programs over a benchmark of Human versus Fungal ortholog groups reported by Szklarczyk et al. (2012). The reported message length and compression values are median statistics across the respective groups. (Note: the $\text{MML}_{\text{Marginal}}$ has N/A values under $\text{I}(\mathcal{A})$ and $\text{I}(\mathbf{S}, \mathbf{T}|\mathcal{A})$ due to the fact, that it gives no specific alignment but a measure integrated over all possible alignments)

	Human remote orthologs of fungal mitochondiral proteins							
	Human vs S.cerevisiae (405 pairs)				*Human vs S.pombe (472 pairs)*			
Program	**%-Hits**	$\text{I}(\mathcal{A}^*)$	$\text{I}(\mathbf{S}, \mathbf{T}\|\mathcal{A}^*)$	Compression	**%-Hits**	$\text{I}(\mathcal{A}^*)$	$\text{I}(\mathbf{S}, \mathbf{T}\|\mathcal{A}^*)$	Compression
ClustalW	71.85	117.4	2678.1	82.6	74.58	110.3	2575.0	77.1
CONTRAlign	71.11	117.8	2679.9	75.7	72.03	108.8	2573.6	70.7
KAlign	73.83	134.3	2639.3	84.9	74.79	24.8	2533.1	81.8
MAFFT	70.79	163.9	2622.1	81.8	71.91	150.9	2535.7	76.7
MUSCLE	73.83	136.5	2639.3	86.1	76.06	129.8	2539.1	84.9
ProbCons	70.12	143.5	2639.3	78.9	71.61	130.4	2543.9	75.9
TCoffee	69.14	141.8	2640.9	76.5	71.19	130.8	2544.4	71.1
$\text{MML}_{\text{Optimal}}$	79.26	119.1	2666.5	93.7	80.51	109.5	2548.9	94.2
$\text{MML}_{\text{Marginal}}$	94.57	N/A	N/A	125.0	94.49	94.49	N/A	117.9

Table 5.3: Comparison of various alignment programs over the 10,250 *twilight zone* pairs from SABmark (Van Walle et al., 2005)

Program	**%-Hits**
ClustalW	1.7951
CONTRAlign	2.3512
KAlign	2.4878
MAFFT	1.8187
MUSCLE	2.4976
ProbCons	1.6683
TCoffee	1.6390
$\text{MML}_{\text{Optimal}}$	4.1399
$\text{MML}_{\text{Marginal}}$	34.9951

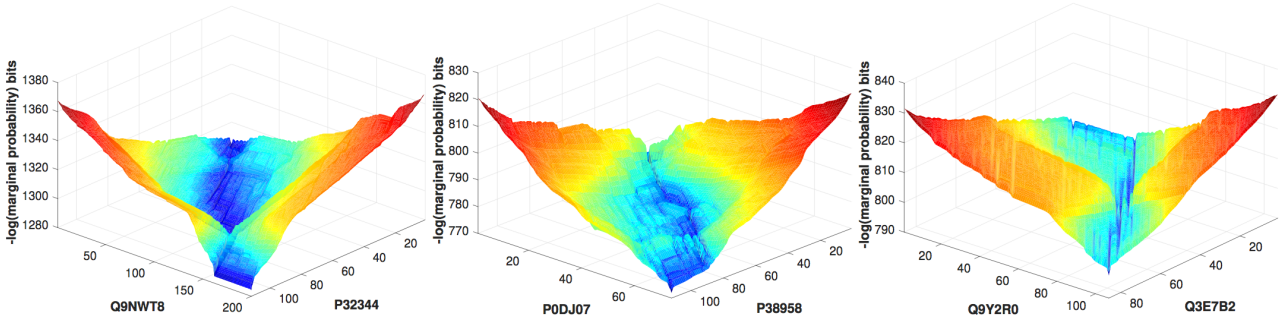
Figure 5.7 presents a visualisation of the marginal probability landscapes for some selected ortholog proteins between humans and yeast, listed in the fungal mitochondrial remote ortholog dataset.

Table 5.3 provides the performance over the challenging *twilight zone* benchmark. This is a difficult dataset for most programs, especially for those that rely on reporting a single optimal alignment. As it can be seen, methods that rely on finding the best alignment under their respective criteria fare far worse than the MML protein alignment model that estimates the marginal probability of the evolutionary relationship between two sequences, and ascertains its statistical significance with the *null model*. $\text{MML}_{\text{Marginal}}$ alignment model is able to identify a substantially greater number of hits (34.9%). A distant second is the $\text{MML}_{\text{Optimal}}$ model (4.1%), followed by MUSCLE (2.5%).

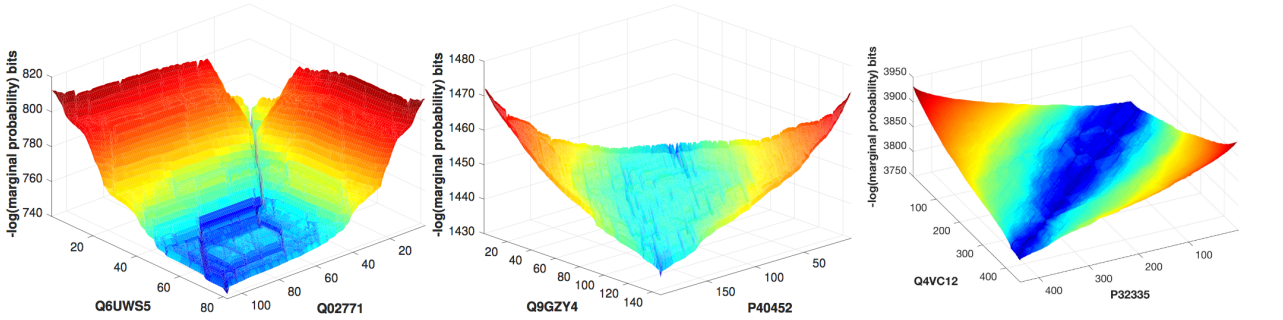
The insights and results presented here establish the ability of the MML protein alignment framework to answer the question *"Are two proteins related? If so, how exactly are they related?"*. They signify the benefit of quantifying the amino acid substitutions and gaps ***in concert*** when aligning two protein sequences. Chapter 6 extends these insights through an improved substitution modelling approach. While this section concludes the main contribution of this chapter, the next section explores a future utility of the MML estimation method (discussed in §5.3) used to derive the evolutionary-time-parameterised Dirichlet distributions.

5.5 Optimal Binning of the Evolutionary Time Parameter: A Future Direction

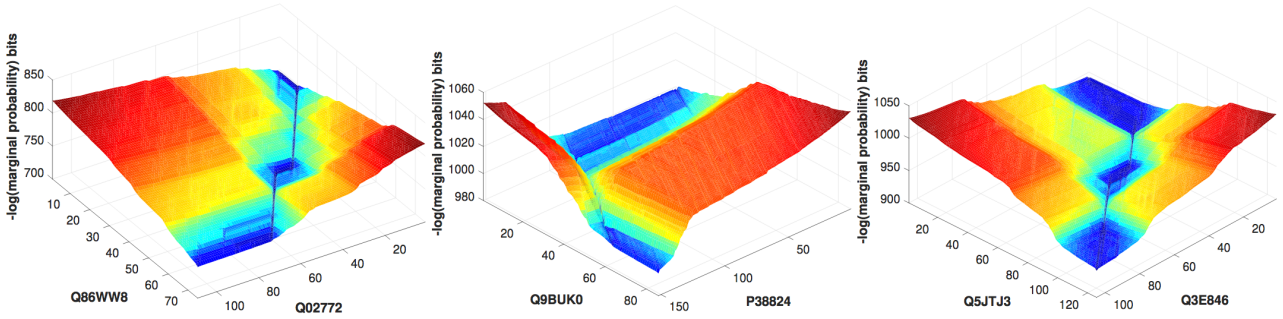
Previously, Dirichlet models were inferred for each integer time t in the range of discrete evolutionary time $[1, t_{\max}]$. However, it is interesting to explore if there exists a partition of this time parameter range such that, all alignments in some bin $(i, j]$ of that range can be optimally explained using a single set of 1-simplex and 2-simplex Dirichlet models, rather than having a separate set of Dirichlet models for each discrete time $t \in (i, j]$. In other words, each bin explains a reasonable, average representation of the three-state machine in terms of 1-simplex and 2-simplex Dirichlet models for all discrete time points within that bin. This problem can be simply formulated as a one-dimensional Dynamic Program described below.



(a) COX24 (Average PAM = 213, compr=14) (b) PET100 (Average PAM = 183, compr=5.1) (c) COA3 (Average PAM = 314, compr=0.9)



(d) PET117 (Average PAM = 91, compr=35) (e) COA1 (Average PAM = 408, compr=-3.1) (f) MSS51 (Average PAM = 300, compr=1.8)



(g) PET191 (Average PAM = 216, compr=29.4) (h) COX23 (Average PAM = 245, compr=16.6) (i) YMR244C-A (Average PAM = 192, compr=36.4)

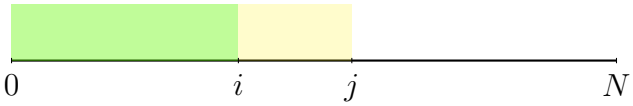
Figure 5.7: A gallery of marginal probability landscapes inferred for nine different human versus yeast ortholog pairs: COX24, PET100, COA3, PET117, COA1,MSS51, PET191, COX23 and YMR244C-A, from the fungal mitochondrial remote ortholog dataset by Szklarczyk et al. (2012) (Note: compr refers to the compression gain of the MML *marginal probability model* in bits, with respect to the *null-model*. Average PAM indicates the optimal evolutionary time t inferred when integrating all possible alignments)

5.5.1 Binning Problem

Given any ordered set $(1, 2, \ldots, N)$, a partition involving k non-overlapping bins/cuts defines a set of indices $j_1, j_2, \ldots, j_{k-1}$ such that $0 < j_1 < j_2 < \cdots < j_{k-1} < N$. These indices define k bins of the form $(0, j_1], (j_1, j_2], (j_2, j_3], \ldots, (j_{k-1}, N]$. Thus, there are $\binom{N-1}{k}$ possible ways to partition the set with exactly k bins/cuts. The minimum number and the maximum number of possible cuts are 0 and $N-1$, respectively. Thus, there are $\sum_{k=0}^{N-1} \binom{N-1}{k} = 2^{N-1}$ total number of possible ways to bin the ordered set $(1, 2, \ldots, N)$.

In the context of this thesis, all discrete points in an evolutionary time range $[1, N]$ form the ordered set of integers. The aim is to partition this range such that, the associated total message length for modelling this time-binning is minimised.

The structure of the optimal solution to this binning problem has an optimal substructure and overlapping sub-problems, enabling the application of Dynamic Programming (Bellman et al., 1954). This problem first requires the evaluation of costs of all feasible bin ranges. Let $\text{Cost}(i, j)$ store the message lengths of explaining alignments whose inferred time parameters are in the range $(i, j]$ (i.e. bin). The optimal substructure of the one-dimensional dynamic program ensures that, the optimal partition up to j can be constructed from the optimal partition up to some i (where $i < j$), by adding to it the $\text{Cost}(i, j)$.



Let $Hist(j)$ represent an one-dimensional array storing the solutions of the optimal partition for each $1 \leq j \leq N$. Initialise $Hist(0) = 0$ denoting the trivial optimal solution of an empty $N = 0$ subproblem. Solutions of the growing optimal subproblem can then be computed using the following DP recurrence:

$$Hist(j) = \min_{\forall 0 \leq i < j} \Big\{ Hist(i) + \text{Cost}(i, j) \Big\}$$

The array $Hist$ is progressively filled until $j = N$. Associated with $Hist$, another array B is maintained, where each $B(j)$ stores the optimal choice of $0 \leq i < j$ for each j. Tracing back using B gives the cut points of the optimal partition of $(0, N]$.

5.5.2 Cost Function for a Bin

Let b denote a bin which represents the range $b = (l, u]$ in the evolutionary time range $[1, t_{\max}]$. The bin b has an associated set of three-state alignment strings $\mathbf{A}_b$. As previously discussed in §5.2.2, Dirichlet models for state $\mathtt{m}$ and $\mathtt{i}$ ($\equiv \mathtt{d}$) over some alignment dataset $\mathbf{A}$ are inferred independently using the objective function Equation 5.3. Accordingly, we can derive optimal Dirichlet models, $\text{Dir}_{\mathtt{m}}(\vec{\alpha}_{\mathtt{m}})$ and $\text{Dir}_{\mathtt{i}}(\vec{\alpha}_{\mathtt{i}})$, over the dataset $\mathbf{A}_b$. The entire bin of alignments $\mathbf{A}_b$ can then be communicated using the inferred models jointly, with a total message length:

$$I(\vec{\alpha}_{\mathtt{m}},\vec{\alpha}_{\mathtt{i}}, \vec{\Theta}, \mathbf{A}_b) = \underbrace{I(\vec{\alpha}_{\mathtt{m}}) + I(\vec{\alpha}_{\mathtt{i}}) + I(\vec{\Theta}_{(\mathtt{m})}|\vec{\alpha}_{\mathtt{m}}) + I(\vec{\Theta}_{(\mathtt{i})}|\vec{\alpha}_{\mathtt{i}})}_{\text{First part}}$$

$$\underbrace{+ I(\mathbf{A}_{b(\mathtt{m})}|\vec{\Theta}_{(\mathtt{m})}, \vec{\alpha}_{\mathtt{m}}) + I(\mathbf{A}_{b(\mathtt{i})}|\vec{\Theta}_{(\mathtt{i})}, \vec{\alpha}_{\mathtt{i}}) + \sum_{\forall \mathcal{A} \in \mathbf{A}} \left[I(|\mathcal{A}|) - log\left(\frac{1}{3}\right) \right]}_{\text{Second part}} \quad \text{bits}$$

Accordingly, cost of some bin $(i, j]$ can be computed and stored in a Cost matrix cell (i, j). Algorithm in Figure 5.8 presents the pseudocode of the one-dimensional Dynamic Programming algorithm for optimal binning of the evolutionary time parameter.

Alternatively, we can also define a full communication of all protein pairs D for which $\mathbf{A}$ have defined alignment hypotheses. Each protein sequence pair $\langle \mathbf{S}_i, \mathbf{T}_i \rangle \in D$ can be encoded according to the *alignment model* discussed in §4.1.1. Recall that, this also requires the statement of the optimal evolutionary time t that determines which amino acid substitution matrix to be used. Since the dataset of the bin represents a range of evolutionary time points, one option is to use $\lfloor \frac{l+u}{2} \rfloor$ (i.e. the mid time-point of the bin: t_{mid}) for this purpose. t_{mid} can be encoded over the uniform probability of $\frac{1}{t_{\max}}$. Accordingly, the total message length now becomes:

$$I(\vec{\alpha}_{\mathtt{m}},\vec{\alpha}_{\mathtt{i}}, \vec{\Theta}, \mathbf{A}_b, D) = I(t_{\text{mid}}) + I(\vec{\alpha}_{\mathtt{m}},\vec{\alpha}_{\mathtt{i}}, \vec{\Theta}, \mathbf{A}_b) + I(D|\vec{\alpha}_{\mathtt{m}},\vec{\alpha}_{\mathtt{i}}, \vec{\Theta}, \mathbf{A}_b) \quad \text{bits} \tag{5.6}$$

```
Data: Cost_matrix
Result: Optimal bin cuts
N ← 1000;
Init Hist[0...N] = 0;
Init B[0...N] = 0;
for j = 1 to j = N do
    min_val ← +∞ ; min_cut_index ← -1;
    for i ← 0 to (j − 1) do
        val ← Hist[i] + Cost(i, j);
        if val ≤ min_val then
            min_val ← val;
            min_cut_index ← i;
        end
    end
    Hist(j) ← min_val;
    B(j) ← min_cut_index;
end
```

Figure 5.8: Pseudocode for the optimal binning of evolutionary time range $[1, t_{\max}]$

This concludes the optimal binning of the evolutionary time parameter (based on a three-state alignment string distribution). It provides us with a new way of modelling the encoding length of the time parameter.

Chapter 6

Modelling Amino Acid Substitutions

"*If you torture the data enough, nature will always confess*"

– Ronald Harry Coase

In the preceding chapters, the Minimum Message Length (MML) protein alignment framework relied on a previously published Markov model of amino acid substitution, namely the PAM model. This chapter completes the inference of the set of statistical models needed for sequence alignment using MML, by inferring a time-parameterised Markov matrix of amino acid substitution from any given alignment benchmark. Specifically, six structural alignment benchmarks (that have been curated using various structural alignment methods) are used to derive an optimal Markov matrix for each of them. Additionally, it allows us to examine nine well-known substitution matrices on those benchmarks; any matrix that does not explicitly model a time-dependent Markov process is converted to a corresponding (base) Markov matrix that does. All fifteen matrices (9 previous and 6 new) are compared by measuring the Shannon information content they yield in explaining each benchmark. Finally, this culminates in a new and overall best performed stochastic matrix, MMLSUM, and its associated three-state machine, whose properties are extensively analysed here.

This chapter describes the material appearing in the following manuscript (which is under communication at the time of writing this thesis):
Sumanaweera, D., Allison, L. and Konagurthu, A.S., 2020. Bridging the Gaps in Statistical Models of Protein Alignment. arXiv preprint arXiv:2010.00855.
URL: https://arxiv.org/abs/2010.00855

6.1 General Approach of Substitution Modelling

Extant proteins have evolved from ancestral proteins over millions of years. A protein's history is not explicit from currently observed protein sequences (Marsh and Teichmann, 2010). An important step in inferring shared relationships between them is to model how one amino acid at any position in some sequence gets substituted by another, as they evolve over a period of time. This forms the basis of amino acid substitution matrices.

The section here introduces the key elements of a typical substitution scoring matrix, complementing what was previously discussed in §2.2.1. It is followed by an overview of history into amino acid substitution modelling and the approaches used to derive scoring matrices.

As discussed in §2.4.1, generating any pairwise sequence alignment between two proteins relies on a fixed 20×20 substitution matrix, to quantify their relationship in terms of amino acid matches. In traditional substitution scoring matrices, typically, each element corresponds to a unique amino acid pair, conveying a (scaled log-odds) score which reflects their degree of interchangeability. That is, each cell $S(i,j)$ allocates a score to an amino acid pair $\langle a_i, a_j \rangle$ based on the log-odds ratio of their probability of being related ($\Pr(a_j)\Pr(a_i|a_j) \equiv \Pr(a_i)\Pr(a_j|a_i)$) to their probability of being unrelated ($\Pr(a_i).\Pr(a_j)$):

$$S(i,j) = c \cdot \log_2 \left(\frac{\Pr(a_j)\Pr(a_i|a_j)}{\Pr(a_i)\Pr(a_j)} \right) = c \cdot \log_2 \left(\frac{\Pr(a_i|a_j)}{\Pr(a_i)} \right) \equiv c \cdot \log_2 \left(\frac{\Pr(a_j|a_i)}{\Pr(a_j)} \right) \tag{6.1}$$

where c is the scaling factor to set the unit of information (e.g. $c = 2$ implies half-bit unit; $c = 3$ implies one-third-bit unit). The choice of c is arbitrary. It is used to ensure that these scores are within a range of integer values when rounded. Nearly all alignment programs employ log-odds scores of substitutions to perform an alignment. Implicitly, the matrix also exhibits (1) an expected change between any amino acid pair (reflecting the average evolutionary time it represents), and (2) a background probability distribution for amino acids.

Any log-odds substitution scoring matrix can be converted into its corresponding *conditional probability* and *joint probability* forms, using the constant c and independent (*null*) probabilities of amino acids: $\Pr(a_i)$ and $\Pr(a_j)$. Note, a conditional probability matrix of substitutions is asymmetric, whereas the joint probability matrix is symmetric.

Deriving these matrices to model amino acid substitutions is often carried out via enumerating observed substitutions within homologous protein sequences. The pioneering work of Dayhoff et al. (1978) introduced a Markov model of substitutions. Since then, many efforts over the past four decades have been aimed at improving these models by leveraging the growth of sequence databases. Their progress is outlined below.

6.1.1 A Brief History

The edit distance between corresponding codons of amino acids was the first basis for scoring matches (Fitch, 1966; Feng et al., 1985; Dayhoff et al., 1978). However this was soon realised to be ineffective at finding distant relationships, since it does not capture amino acid properties well (McLachlan, 1971; Tomii and Kanehisa, 1996). Concurrently, scoring matrices based on physicochemical characteristics arose, affirming atomic composition, polarity, molecular volume and hydrophobicity to correlate well with substitutions (Grantham, 1974; Miyata et al., 1979; George et al., 1990). Better models (both Markov and non-Markov) appeared when protein sequences with known homology became a reliable data source for learning substitution patterns (McLachlan, 1971; Dayhoff et al., 1978; Jones et al., 1992a; Gonnet et al., 1992; Henikoff and Henikoff, 1992; Gonnet and Korostensky, 1999; Kosiol and Goldman, 2005; Huang, 2008; Keul et al., 2017). Early consideration was given to substitution counts (frequencies) amongst closely related proteins. Later on, the inclusion of distantly related proteins and the evolutionary time between sequences became prominent. Shortly, the importance of bringing in the evolutionary conservation of protein structural elements to the substitution modelling process was also acknowledged. This resulted in more matrices based on aspects such as secondary structure context, structural superposition and alignments, pairwise amino acid contacts, surface exposure, torsion angle distributions and potential energy based structural stability (Levin et al.,

1986; Rao, 1987; Risler et al., 1988; Niefind and Schomburg, 1991; Lüthy et al., 1991; Kolaskar and Kulkarni-Kale, 1992; Johnson and Overington, 1993; Miyazawa and Jernigan, 1993; Rice and Eisenberg, 1997; Russell et al., 1997; Dosztanyi and Torda, 2001; Qian and Goldstein, 2002; Qiu and Elber, 2006; Tan et al., 2006; Farheen et al., 2017). Recently, many have explored substitution parameter optimisation approaches (sometimes alongside gap parameter optimisation) rather than depending on observed substitution counts alone (Levin et al., 1986; Kann et al., 2000; Müller and Vingron, 2000; Whelan and Goldman, 2001; Müller et al., 2002; Qian and Goldstein, 2002; Holmes and Rubin, 2002; Hourai et al., 2004; Saigo et al., 2006; Le and Gascuel, 2008; Yamada and Tomii, 2013; Herman et al., 2015). The general approach is to optimise the matrix over a predefined performance criterion such as homology classification, secondary structure prediction accuracy, or tertiary structure similarity. Some have also emphasised the importance of adjusting prevailing scoring matrices to suit sequences with biased amino acid compositions (Yu et al., 2003; Yu and Altschul, 2004).

General purpose amino acid substitution matrices are often viewed as widely useful, well performing and robust (Keane et al., 2006; Le and Gascuel, 2008; Kosiol and Goldman, 2011). However, most of them are learned over mammalian proteins; other unique contexts such as viral proteins may require more specific models (Nickle et al., 2007). Thus there exists various studies on protein-family or species-specific matrices as well (Adachi and Hasegawa, 1996; Adachi et al., 2000; Ng et al., 2000; Müller et al., 2001; Dang et al., 2010). Nevertheless, Keane et al. (2006) concludes that the decision on a substitution matrix should not be based on its source or construction method. Some also advocate ensemble models, arguing that different models could be applicable to different sequence pairs instead of a single, fixed substitution model (Huelsenbeck et al., 2008).

6.1.2 Existing Markov Models of Substitution

The first and the most notable effort of amino acid substitution modelling was made by Dayhoff et al. (1978) via the PAM (Point Accepted Mutation) series of matrices. Underlying the PAM is a discrete Markov model. It was estimated using a set of phylogenetic trees constructed on proteins from 71 closely related families, with at least 85% identity between the amino acid sequences. It was the first work to propose the PAM (time) unit, defined as the state of a Markov matrix showing a 1% expected change in amino acids. (This is also referred to as the PAM-1 matrix). Accordingly, the substitution matrix for an evolutionary time parameter t is given by the matrix PAM-t, computed as the t^{th} power of PAM-1. Improved versions (such as the JTT matrix (Jones et al., 1992a) and several more (Gonnet et al., 1992; Gonnet and Korostensky, 1999; Kosiol and Goldman, 2005)) were also published later on. One pragmatic argument commonly made against PAM is that, its construction was done using a limited set of (then) available protein sequences. Later models focused on rectifying this. Separately, based on PAM, Altschul (1993) introduced a substitution scoring scheme called the 'All-PAM scoring system' which is sensitive to all detectable evolutionary distances of time.

Other directions of Markov model estimation include Bayesian approaches, methods for deriving models from a non-Markov series, and also the ways of capturing codon level bias and protein-level selective constraints together (Yang et al., 1998; Devauchelle et al., 2001; Veerassamy et al., 2003; Arvestad, 2006; Kosiol and Goldman, 2011; Ndhlovu et al., 2015). As for the family of continuous-time Markov models, VT (Müller and Vingron, 2000) and VTML (Müller et al., 2002) matrices were resulted from an approach called 'matrix resolvent method' and a Maximum Likelihood (ML) method over sequence alignments. The WAG matrix (Whelan and Goldman, 2001) is another model that employs an ML approach over inferred phylogenetic trees on globular protein sequence families. Le and Gascuel (2008) presented the LG model as

an improvement on WAG, covering a larger and more diverse set of protein family alignments while also considering rate variation across sites. Holmes and Rubin (2002) came up with an ML based Expectation-Maximisation (EM) algorithm for inferring substitution rate matrices from multiple sequence alignments, generalisable to derive matrices for varying sites/structural contexts. Despite claims that amino acid sequence evolution is non-Markovian (Bennet et al., 1994; Kosiol and Goldman, 2011), a Markov model remains the widely-used form to describe the amino acid substitution process during divergent evolution.

How the PAM (Point Accepted Mutation) Series was Derived

Dayhoff et al. (1978) published the PAM-1 stochastic matrix via the Atlas of Protein Sequence and Structure in 1978. A point accepted mutation is an amino acid substitution which is accepted by nature. This matrix hypothesises probabilities for all possible point accepted mutations, defined for a period of evolutionary time where 1% expected change is seen (i.e. PAM-1 unit of time, as mentioned previously). It was inferred over phylogenetic tree based multiple alignments of 85% or more identical sequences from 71 families under 34 superfamilies, imposing the following assumptions:

1. The dataset is sufficient in representing single, direct point mutations (without any unobserved, intermediate mutations – E.g. $A \to B$ is a single-step mutation, and there was no $A \to X \to \ldots Y \to B$ for any X, Y)

2. The applied phylogenetic tree method infers reasonable common ancestral sequences and a multiple sequence alignment

3. Mutations are site-independent

4. The transition rate of any amino acid type/position is a constant

Matrix construction goes as follows. A symmetric, raw substitution count matrix A is derived using generated phylogenetic trees, by comparing observed sequences with their corresponding (inferred) ancestral sequences. Each $A(i, j)$ denotes the number of observed mutations between amino acid i and amino acid j. This is symmetric since the direction of mutations is not taken into account.

Next, they define the term *relative mutability* of an amino acid a_j denoted by m_j, as proportional to the probability of a_j changing during a small evolutionary time interval (i.e. one PAM unit of time): $\Pr(a_j \to *)$. A simple estimate for m_j is the proportion of changes from a_j to any other amino acid $(*)$, out of all a_j occurrences. Dayhoff et al. (1978) computes a weighted estimation for this over all phylogenetic trees, where a weight factor is defined as an estimator of the exposure to evolution (i.e. the ratio between the total number of mutations and the number of amino acids for a branch/pair) (Kosiol and Goldman, 2005). Following the simple estimation, we have:

$$\Pr(a_j \to *) = l \cdot m_j = l \cdot \frac{\sum_{\forall i \neq j} A(i, j)}{\sum_{\forall i} A(i, j)}$$

where l is the constant rate parameter which acts as the proportional constant for m_j to correct for any possible, yet unknown indirect mutations within observed substitutions.

Further, the *relative frequency* of an amino acid j (denoted by f_j) is defined as its probability of occurrence. Again, a simple estimate of the proportion of the amino acid a_j over all amino

acids can be taken for f_j:

$$\Pr(a_j) = \frac{\sum_{\forall i} A(i,j)}{\sum_{\forall x,y} A(x,y)}$$

This is again a weighted estimate by Dayhoff et al. (1978).

Finally, the product rule is applied to compute the mutation probability of an amino acid a_j changing into an amino acid a_i $(i \neq j)$:

$$\begin{aligned} \Pr(a_j \rightarrow a_i) &= \Pr(a_j \rightarrow *) \cdot \Pr(a_j \rightarrow a_i \mid a_j \rightarrow *) \\ &= l \cdot m_j \left(\frac{A(i,j)}{\sum_{\forall x,y(x \neq y)} A(x,y)} \right) \end{aligned}$$

Accordingly, a 20×20 conditional probability matrix (also known as mutation probability matrix) M can be filled with all non-diagonal elements computed as above. A diagonal element $M(j,j)$ is simply the probability of amino acid j remaining unchanged, and can be estimated by $(1 - l.m_j)$.

The rate constant l is derived such that the M matrix corresponds to 1% average substitutions (0.01 expected change). Note that, the probability of observing a change in any amino acid is given by the below weighted sum:

$$\Pr(\text{observing a change}) = \sum f_j \cdot l \cdot m_j = 0.01$$

In that way, l is selected to realise one PAM unit of time.

The PAM-1 matrix M is represented as a left stochastic matrix where each column adds up to 1. A cell $M(i,j)$ gives the probability of an amino acid a_j changing to an amino acid a_i, given 1% expected number of direct amino acid mutations at $t = 1$. Consequently, $M^t(i,j)$ cell (i.e. in the PAM-t matrix) gives the same probabilities, yet over a Markov chain of (length) t transitions from a_j to a_i. The properties of such a Markov model are discussed in §6.2. The conditional probability matrix M^t is converted into a scaled log-odds scoring matrix S, rounded to the nearest integer as follows:

$$S(i,j) = \texttt{round}\left(10 \times \log_{10}\left(\frac{M^t(i,j)}{f_i}\right)\right)$$

George et al. (1990) have observed that, despite several shortcomings such as: (1) the insufficiency of observed substitution data (i.e. due to having no observations for 35 amino acid exchanges out of the 400 possibilities), (2) a bias towards soluble proteins, (3) the absence of types such as membrane proteins at the time of publication, (4) unreliable ancestral sequence inferences, (5) circular dependency on multiple sequence alignments, and (6) the assumption of a constant rate of mutation, PAM (specifically PAM-250) is still widely used; mostly it comes as a default choice in many sequence alignment tools.

6.1.3 Existing Non-Markov Models of Substitution

The widely-used BLOSUM (Henikoff and Henikoff, 1992) is the most notable in the domain of non-Markov matrix series. Each matrix in the series was inferred over a set of multiple *ungapped* alignments of conserved regions in protein families from Henikoffs' BLOCKS database (Pietrokovski et al., 1996). BLOSUM-n is derived by enforcing a clustering threshold of n, ensuring that the matrix represents amino acid substitutions only from protein pairs that have at

most $n\%$ sequence identity. While BLOSUM remains widely-accepted for sequence alignment, there are other attempts to improve the model further. For instance, Kann et al. (2000) took BLOSUM-62 to be a starting point for an EM based numerical optimisation method, resulting in a new matrix called OPTIMA. They optimised its performance based on an average confidence measure of remote sequence homology detection with the affine gap penalty function. An MCMC approach was employed by Herman et al. (2015) for maximising matrix likelihood over multiple sequence alignments, using a Gamma prior on substitution counts based on BLOSUM-62 values. Recently, a new matrix family called PFASUM (Keul et al., 2017) was obtained over manually curated Pfam seed multiple sequence alignments using the same clustering technique as BLOSUM, however not only taking the conserved regions but also the gap rich columns into account. Separately, Yamada and Tomii (2013) explored a new line of thought through Principal Component Analysis (PCA) of nine existing matrices (including several from BLOSUM and VTML family), for sampling a new matrix called MIQS from the most sensitive region of the corresponding PCA 3D subspace.

Amongst the structure based substitution models is a well-known matrix by Johnson and Overington (1993), derived over a set of complete structural alignments by accounting not only for the most similar portions but also for variable regions amongst protein structures. STROMA matrix (Qian and Goldstein, 2002) has been inferred through an iterative optimisation of substitution scores and gap penalties together, over the structural similarity defined in terms of the average Root-mean-square deviation (RMSD) values computed on a set of rigid-body superposed, distant proteins. Qian and Goldstein (2002) claim that the matrix is indifferent to any gap parameter choice, mainly due to most gap penalty combinations resulting in low RMSD values.

Nevertheless, despite the common presumption that structure gives more information on amino acid substitutions, Russell et al. (1997) observed that their scoring matrix derived over structural alignments was in fact closer to PAM-250 and BLOSUM-62.

Additionally, some have examined how stability and potential energy of a protein tertiary structure relate to substitution propensities (Miyazawa and Jernigan, 1993; Dosztanyi and Torda, 2001). The idea follows the fact that a physicochemically similar amino acid substitution is unlikely to destabilise the native structure.

How the BLOSUM Series was Derived

BLOSUM (BLOck SUbstitution Matrix) (Henikoff and Henikoff, 1992) has been inferred over the BLOCKS database (Pietrokovski et al., 1996). A block represents an ungapped, conserved region across a set of related proteins, presented in the form of a multiple sequence alignment. Henikoff and Henikoff (1992) utilised more than 2000 blocks obtained over hundreds of protein families/groups. The matrix construction starts with preparing a table of amino acid pair frequencies (counts) for each block.

Let some block be a set of X aligned sequences with length L, represented by a table where each row corresponds to a sequence. The pair frequency table records column-wise counts for each pair. For instance, if a column in alignment has $\langle 2N, 7A, 1S \rangle$, there are ${}^{2}\mathrm{C}_{2}$ $\langle N, N \rangle$ pairs, ${}^{7}\mathrm{C}_{2}$ $\langle A, A \rangle$ pairs, 7×2 $\langle A, N \rangle$ pairs, 1×2 $\langle S, N \rangle$ pairs, and 7×1 $\langle A, S \rangle$ pairs. The total number of pairs from a column is $\frac{X(X-1)}{2}$. As a result, the total number of pairs from the block is $\frac{LX(X-1)}{2}$. Next, all counts are summed up across all blocks, giving the symmetric, 20×20 total count matrix F. A frequency count f_{ij} of a pair $\langle a_i, a_j \rangle$ is normalised to compute its joint probability of occurrence: $q_{ij} = \frac{f_{ij}}{\sum_{\forall i,j} f_{ij}}$. Meanwhile, their expected probability of occurrence e_{ij} of the pair is computed in terms of individual amino acid occurrence probabilities (p_i and

p_j) as:

$$e_{ij} = \begin{cases} 2p_i p_j & \text{when } i \neq j \\ p_i p_j & \text{when } i = j \end{cases}$$

Here, the pair is viewed as two positions representing a bidirectional substitution (thus $e_{ij} = p_i.p_j + p_j.p_i$ for $i \neq j$) unless it represents no substitution ($i = j$). The occurrence probability p_i of amino acid a_i is computed as:

$$p_i = q_{ii} + \sum_{\forall i \neq j} \frac{1}{2} q_{ij}$$

Note: for an empty pair $\langle \_, \_ \rangle$, the probability of a_i occurrence in any of the two positions is: $\Pr(\text{Choosing the position for } a_i) \cdot \Pr(a_i \text{ occurring jointly with } a_j)$.

With the above information, a scoring matrix S is formed by calculating the log odds ratio between q_{ij} and e_{ij} as:

$$S(i,j) = \texttt{round}\left(2 \times \log_2\left(\frac{q_{ij}}{e_{ij}}\right)\right) \quad \text{half bits}$$

Following this method, Henikoff and Henikoff (1992) produced a series of matrices that represents different evolutionary times. Their approach was to cluster the sequences within a block under a *clustering percentage* n such that, all sequence pairs considered for substitution modelling within the block only represent sequence identity percentages less than $n\%$. The resultant is a BLOSUM-n matrix. This means, sequences that are at least $n\%$ identical are clustered together and considered as a single, average sequence when taking their pair frequency count contributions into account for preparing the count matrix. One approach to clustering is by first appointing all sequences in the block as nodes of a fully connected graph (Durand, 2015). Each edge has its weight taken to be the percentage identity between the corresponding pair. Then, all edges with a weight less than $n\%$ are removed. The resultant connected components are taken as clusters, thus averaging the pair counts across the clustered sequences.

6.2 Markov Model Properties

This section presents the associated mathematical descriptions and internal mechanics of Markov models that enable a coherent representation of how sequences evolve with a notion of evolutionary time. They are essential in building up the contributions of this chapter.

Amino acid substitutions are often explained in the form of a *time homogeneous* and *reversible* Markov chain (Stewart, 1994; Norris, 1998), where the finite state space is the set of 20 amino acid types. It assumes all substitutions to be generating from a *memoryless* state transition process, with independent and identically distributed amino acid sites. This is also an *irreducible* and *aperiodic* chain, since any state can be reached from any other state in finite time, and with a period of 1.[1] The description of the chain's behaviour depends on whether the time is treated as a discrete or continuous variable.

[1] A Markov chain is periodic if the number of single step transitions required for revisiting any state is a multiple of an integer greater than 1 (Stewart, 1994).

6.2.1 Discrete Time Markov Chain

In a Discrete Time Markov Chain (DTMC), states are observed in discrete time steps. Given a discrete time step sequence $\{0, 1, 2, .., t\}$ and a set of state observations $\{x_0, x_1, .., x_t\}$ at each time step, the next state x_{t+1} at time $t+1$ depends only on the current state x_t:

$$\Pr(x_{t+1}|x_0, x_1, \ldots, x_t) = \Pr(x_{t+1}|x_t)$$

The conditional probability of the next state x_{t+1} given the current state x_t corresponds to a single step transition between the two states as observed after one discrete unit of time from the current time t. The common standard in substitution modelling is to take this unit as the time taken for observing a 1% expected change, originally proposed by Dayhoff et al. (1978) with one PAM unit of time (as discussed in the previous section).

All 400 conditional probabilities are conveyed through a 20×20 base probability matrix $\mathbf{M}$ (also known as a stochastic matrix or Markov matrix), where each cell $\mathbf{M}_{ij}$ refers to the probability of an amino acid a_j changing to an amino acid a_i in one unit of time ($0 \leq \mathbf{M}_{ij} \leq 1$). Each column vector of this matrix is an $\mathbb{L}_1$-normalised, 20-state probability vector in a unit 19-simplex ($\sum_{\forall j} \mathbf{M}_{ij} = 1$).

While $\mathbf{M}(1) = \mathbf{M}$ is a *base matrix* which corresponds to all transitions occurring in $t = 1$ unit of time, let us define $\mathbf{M}(t)$ as the substitution probability matrix after t steps of time. The *time homogeneous* property of the Markov chain ensures a transition matrix $\mathbf{M}$ that is invariant of time. Therefore, $\mathbf{M}(t)$ can be obtained by raising the base matrix $\mathbf{M}(1)$ to the power of t:

$$\mathbf{M}(t) = (\mathbf{M}(1))^t$$

At $t = 0$, we expect no change in any amino acid state, informed by $\mathbf{M}(0)$ (i.e. the identity matrix $\mathbf{I}$). As time progresses, an amino acid position could stay in the same state for some time (i.e. holding time) until a transition to a different state. Since the process is captured in discrete time units, this holding time is geometrically distributed.

Another important property of an *irreducible* and *aperiodic* Markov chain is that, the following equation has a unique solution:

$$\vec{\pi} \cdot \mathbf{M} = \vec{\pi}$$

where $\vec{\pi} = [\pi_1, \pi_2, ..., \pi_{20}]$ is the 20-dimensional stationary distribution vector of the Markov chain such that, $\Pi = [\vec{\pi}^T, \vec{\pi}^T, ..., \vec{\pi}^T]_{20\times 20} = \lim_{t\to\infty} \mathbf{M}(t)$. The probability π_i is the stationary probability (i.e. equilibrium probability) of an amino acid a_i. This property explains the evolution of the Markov chain. As the system evolves, the conditional probabilities reach the corresponding stationary probabilities, known as the equilibrium state.

Further, the system also satisfies the following equation (known as the detailed balance equation):

$$\pi_j \mathbf{M}_{ij}(t) = \pi_i \mathbf{M}_{ji}(t) \tag{6.2}$$

implying *reversibility*. This reflects the product rule form of the Bayes theorem (explained in §3.1.2). Accordingly, the above value gives the joint probability of observing an amino acid pair $\langle a_i, a_j \rangle$ as related.

6.2.2 Eigen Decomposition

Eigen decomposition of the stochastic matrix $\mathbf{M}$ enables an efficient computation of $\mathbf{M}(t)$. Any diagonalisable, square matrix A of size $n \times n$ can be decomposed into a canonical form

represented by its eigenvalues and eigenvectors. A scalar λ is an eigenvalue for A, if there exists a non-zero vector $\vec{u} \in \mathbb{R}^n$ that satisfies:

$$\begin{aligned} A\vec{u} &= \lambda\vec{u} \\ \implies (A - \lambda\mathbf{I})\vec{u} &= \vec{0} \end{aligned}$$

Then, $\vec{u}$ is called an eigenvector of the matrix. This reflects the application of a linear transformation described by A on $\vec{u}$ that merely scales $\vec{u}$ along its original direction by a factor of λ. Finding the eigenvalues and the respective eigenvectors of A involves solving the below characteristic polynomial:

$$|A - \lambda\mathbf{I}| = 0$$

resulting in n number of eigenvalues (represented by the diagonal matrix Λ) and their corresponding eigenvectors (represented by the matrix of column eigenvectors $\mathbf{U}$). The eigen decomposition theorem states that the matrix A can be decomposed as:

$$A = \mathbf{U}\Lambda\mathbf{U}^{-1}$$

Applying this insight to $\mathbf{M}(t) = (\mathbf{M}(1))^t$, we get:

$$\mathbf{M}(t) = \left(\mathbf{S}\Lambda\mathbf{S}^{-1}\right)^t = \mathbf{S}\Lambda^t\mathbf{S}^{-1} \tag{6.3}$$

where $\mathbf{S}$ and $\mathbf{\Lambda}$ are the eigenvector and (diagonal) eigenvalue matrices of $\mathbf{M}(1)$.

The spectrum of the stochastic matrix (i.e. the set of all eigenvalues $\{\lambda_1, \lambda_2, ..., \lambda_{20}\}$ in their descending order) is real and positive valued. The largest eigenvalue ($\lambda_{\max} = \lambda_1$) is 1. (This is also known as the *Perron-Frobenius* eigenvalue – every other $\lambda_i < \lambda_{\max}$). The eigenvector associated with $\lambda_{\max}$ corresponds to the stationary distribution. The matrix rate of convergence to the equilibrium state is controlled by the difference between $\lambda_{\max}$ and λ_2 (i.e. *spectral gap*) as $t \to \infty$ asymptotically (Berestycki, 2016). The time taken to reach the stationary distribution is referred to as the *mixing time*. The Perron-Frobenius theorem (Perron, 1907; Frobenius et al., 1912) explains how the distance to equilibrium state decays exponentially for irreducible finite Markov chains (Hough, 2003; Berestycki, 2016). The mixing time is inversely proportional to the spectral gap. This can be intuitively reasoned without going into a formal mathematical proof. When exponentiating the matrix to a higher power (order), the eigenvalues get exponentiated each time (according to the decomposition given in Equation 6.3). At equilibrium, all eigenvalues have reached 0 except for $\lambda_{\max}$ which remains a constant. This means, all eigenvalues except for $\lambda_{\max}$ control the convergence of the matrix. The most restricting eigenvalue is the second largest (λ_2). If it is very close to 1, it takes more time to reach 0. See Figure 6.1 for an example illustration.

Stationary Distribution of a Transition Probability Matrix

Let A be an $n \times n$ stochastic matrix with an eigenvector matrix $\mathbf{U} = \{u_1, u_2, \ldots, u_n\}$, and eigenvalues $\{\lambda_{\max} = \lambda_1, \lambda_2, \ldots, \lambda_n\}$ in their decreasing order. Since $\mathbf{U}$ forms a basis for $\mathbb{R}^n$, any vector $\vec{x} \in \mathbb{R}^n$ (with $\sum_{i=1}^{n} x_i = 1$) can be represented as a linear combination of the eigenvectors in $\mathbf{U}$:

$$\vec{x} = a_1\vec{u_1} + a_2\vec{u_2} + ... + a_n\vec{u_n}$$

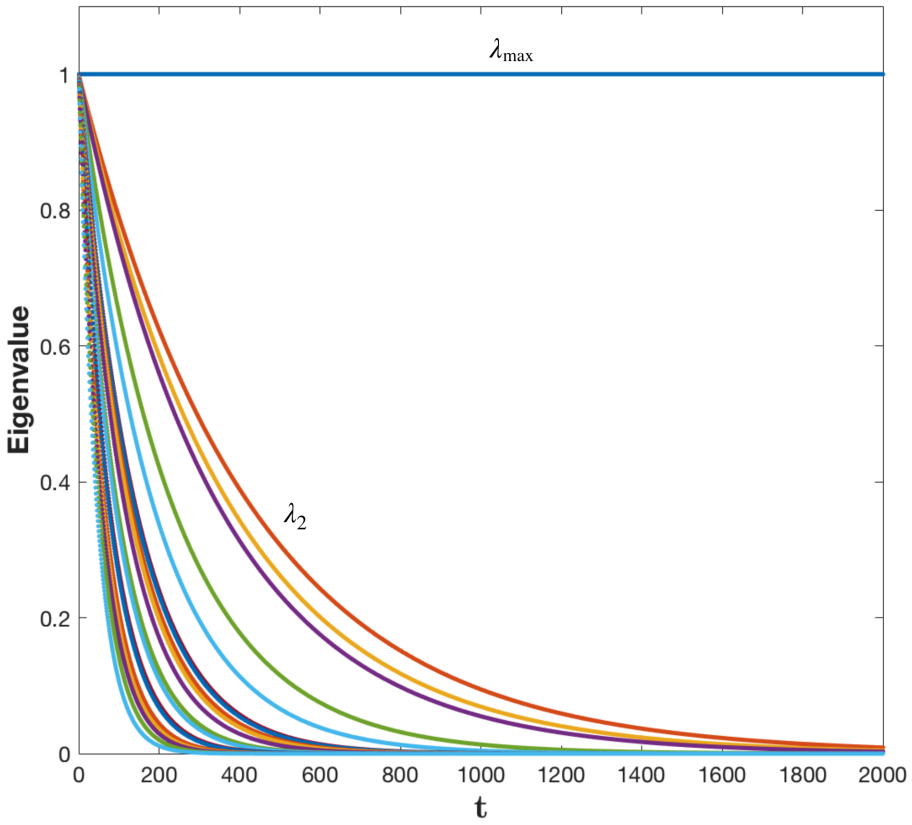


Figure 6.1: How the eigen spectrum of PAM (Dayhoff et al., 1978) transition probability matrix $\mathbf{M}(t)$ varies with matrix power t. Note that the largest eigenvalue $\lambda_{\max}$ remains 1, while others reach 0. The second largest eigenvalue (λ_2) takes the longest time to reach 0.

Considering the transformation A applied to $\vec{x}$,

$$\begin{aligned} A\vec{x} &= a_1 A\vec{u_1} + a_2 A\vec{u_2} + ... + a_n A\vec{u_n} \\ &= a_1\lambda_1\vec{u_1} + a_2\lambda_2\vec{u_2} + ... + a_n\lambda_n\vec{u_n} \end{aligned}$$

($\because \forall i, Au_i = \lambda_i u_i$ as described in 6.2.2). This transformation is equivalent to computing the marginal probability vector over a state space Ω after one step transition, given a prior probability vector $\vec{x}$ and a conditional probability matrix A. For instance, the marginal probability of any state $\mathbf{X_i} \in \Omega$ after a single-step state transition is:

$$\Pr(\mathbf{X_i} \text{ after one step transition}) = \sum_{\forall \mathbf{Y_j} \in \mathbf{\Omega}} \Pr(\mathbf{X_i}|\mathbf{Y_j})\Pr(\mathbf{Y_j})$$

To compute the marginal probability vector $\vec{x_t}$ after t steps of transition, we can apply A transformation on $\vec{x}$ iteratively for t times, yielding, $\vec{x_1} = A\vec{x}, \vec{x_2} = A\vec{x_1}, \vec{x_3} = A\vec{x_2}$ all the way up to $\vec{x_t} = A\vec{x}_{t-1}$. Ultimately, $\vec{x_t} = A^t\vec{x}$. Let us now consider the transformation A^t applied to

$\vec{x}$:

$$\begin{aligned} A^t\vec{x} &= a_1A^t\vec{u_1} + a_2A^t\vec{u_2} + ... + a_nA^t\vec{u_n} \\ &= a_1A^{t-1}A\vec{u_1} + a_2A^{t-1}A\vec{u_2} + ... + a_nA^{t-1}A\vec{u_n} \\ &= a_1A^{t-1}\lambda_1\vec{u_1} + a_2A^{t-1}\lambda_2\vec{u_2} + ... + a_nA^{t-1}\lambda_n\vec{u_n} \\ &= a_1\lambda_1A^{t-2}A\vec{u_1} + a_2\lambda_2A^{t-2}A\vec{u_2} + ... + a_n\lambda_nA^{t-2}A\vec{u_n} \end{aligned}$$

... continued for t times

$$\implies A^t\vec{x} = a_1\lambda_1^t\vec{u_1} + a_2\lambda_2^t\vec{u_2} + ... + a_n\lambda_n^t\vec{u_n}$$

When $t \to \infty$, all $\lambda_i < 1$ involved terms in the above expression goes to 0, while the $\lambda_{\max} = 1$ term remains the same. Accordingly, $A^t\vec{x}$ approaches $a_1\vec{u_1}$ (known as the *steady state*). This is the stationary distribution of the Markov model.

6.2.3 Continuous Time Markov Chain

Continuous Time Markov Chain (CTMC) generalises DTMC to real-valued time. A transition from state j to state i is now considered as occurring in some infinitesimal Δt time instead of one unit of time. The holding time is no longer discrete, hence exponentially distributed. A constant, 20×20 instantaneous rate matrix $\mathbf{Q}$ describes how the probability matrix changes with time, arising from the following derivation.

Denote the probability of a state change after an instantaneous time Δt as $\mathbf{M}(\Delta t)$. The probability matrix at a certain time t depends on the rate at which it changes:

$$\frac{d}{dt}\,\mathbf{M}(t) = \lim_{\Delta t \to 0} \frac{\mathbf{M}(t + \Delta t) \;-\; \mathbf{M}(t)}{\Delta t}$$

This is further simplified using the Chapman-Kolmogorov forward and backward equations which provide for any time s and t, $\mathbf{M}(s + t) = \mathbf{M}(s) \cdot \mathbf{M}(t) = \mathbf{M}(t) \cdot \mathbf{M}(s)$. Accordingly, $\mathbf{M}(t + \Delta t) = \mathbf{M}(t) \cdot \mathbf{M}(\Delta t)$, resulting in:

$$\begin{aligned} \frac{d}{dt}\,\mathbf{M}(t) &= \mathbf{M}(t)\; \underbrace{\lim_{\Delta t \to 0} \frac{\mathbf{M}(\Delta t) \;-\; \mathbf{M}(0)}{\Delta t}}_{\mathbf{Q}} \\ &= \mathbf{M}(t)\;.\mathbf{Q} \end{aligned}$$

Each cell (i, j) in $\mathbf{Q}$ refers to the rate at which an amino acid a_j changes to an amino acid a_i. Further simplification gives the following relationship between $\mathbf{M}(t)$ and $\mathbf{Q}$:

$$\begin{aligned} \int \frac{1}{\mathbf{M}(t)} d\,\mathbf{M}(t) &= \int \mathbf{Q}\; dt \\ \implies \log(\,\mathbf{M}(t)\,) &= \mathbf{Q}t \end{aligned}$$

This results in the below important equation.

$$\mathbf{M}(\mathrm{t}) = \mathrm{e}^{\mathbf{Q}t} \tag{6.4}$$

If we consider a process capture from $t = 0$ to $t = \Delta t$, $\mathbf{M}(\Delta t)$ is given by $e^{\mathbf{Q}\Delta t}$. By expanding this relation via the Maclaurin series and taking the first order approximation, we get:

$$\mathbf{M}(\Delta t) \ = \mathbf{I} + \sum_{k=1}^{\infty} \frac{(\mathbf{Q}\Delta t)^k}{k!} = \mathbf{I} + \mathbf{Q}\Delta t + O(\Delta t)$$

This relation informs the state transition probability values at $t = \Delta t$. Altogether, $\mathbf{Q}$ obtains the following form. Let $\mathbf{Q}_{ij}$ denote a cell (i, j) in $\mathbf{Q}$ that gives the transition rate between any two states, a_i and a_j. Then,

$$\mathbf{Q}_{ij} \geq 0 \quad \text{and} \quad \mathbf{Q}_{jj} = -\sum_{\forall i \neq j} \mathbf{Q}_{ij}$$

Each column of this rate matrix adds upto 0. The system still reaches the stationary distribution $\vec{\pi}$ as it evolves, and it is the unique solution to the equation $\vec{\pi}\mathbf{Q} = \vec{0}$. Also the detailed balance equation is satisfied as $\pi_i \mathbf{Q}_{ji} = \pi_j \mathbf{Q}_{ij}$ (holding reversibility). This also gives rise to a symmetric exchangeability matrix $\mathbf{E}$ where $\mathbf{E}_{\text{ij}} = \frac{\mathbf{Q}_{ij}}{\pi_i} = \frac{\mathbf{Q}_{ji}}{\pi_j}$, resulting in:

$$\mathbf{Q} = \ \mathbf{E}\ \Pi \tag{6.5}$$

where Π is the diagonal matrix with values in $\vec{\pi}$ along the diagonal (Whelan and Goldman, 2001).

Overall, the CTMC process is described using the instantaneous rate matrix rather than the base stochastic matrix that refers to 1 unit of time. The relationship between the stochastic matrix $\mathbf{M}$ and rate matrix $\mathbf{Q}$ is straightforward due to Equation 6.4, enabling a two-way conversion between them given the knowledge of time t (Kishino et al., 1990; Kosiol and Goldman, 2005). Following the eigen decomposition

$$\mathbf{Q} = \mathbf{U} \begin{bmatrix} \lambda_1 & & & \phi \\ & \lambda_2 & & \\ & & . & \\ \phi & & & \lambda_{20} \end{bmatrix} \mathbf{U}^{-1}$$

where $\mathbf{U}$ is the eigenvector matrix, a direct relationship

$$\mathbf{M}(\text{t}) \ = \mathbf{U} \begin{bmatrix} e^{\lambda_1 t} & & & \phi \\ & e^{\lambda_2 t} & & \\ & & . & \\ \phi & & & e^{\lambda_{20} t} \end{bmatrix} \mathbf{U}^{-1}$$

exists according to the Equation 6.4, that supports an instantaneous rate matrix $\mathbf{Q}$ for any stochastic matrix $\mathbf{M}(t)$ with a valid logarithm. CTMC rate matrix estimation for amino acid substitution is not trivial due to the difficulty in finding sequence relationships that truly reflect an infinitesimal amount of time. Any DTMC model such as PAM also implicitly carries a rate matrix that defines its CTMC version. Kosiol and Goldman (2005) explored rate matrix derivation for PAM when t approaches 0 upto the numerical limits in modern day systems.

Expected Change of a Markov Matrix

The expected change of a stochastic matrix is defined as the probability of observing a change in any state (on average). Since the matrix diagonal represents all self-state transitions, the probability of no observed change in any state on average can be computed over the diagonal, and be used to compute the opposite event. Accordingly, for any substitution probability

matrix $\mathbf{M}(t)$:

$$\text{Expected change} = 1.0 - \sum_{\forall j} \pi_j \mathbf{M}_{jj}(t)$$

In contrast, the instantaneous rate of change of some state j is given by $-\mathbf{Q}(j,j)$. Accordingly, the expected rate of change is calculated as:

$$\text{Expected rate of change} = -\sum_{\forall j} \pi_j \mathbf{Q}_{jj}$$

6.2.4 Converting a Non-Markov Matrix into a Markov Matrix

A non-Markov substitution model represents a random process of amino acid substitutions which does not adhere to the Markov property nor represent protein sequence evolution as a Markov chain of events. Such model either gives a single substitution matrix or a series of evolutionary-time-parameterised matrices that have been estimated independently (§6.1.3 gives an overview to the existing matrices/series as such).

Given a set of Markov substitution models and non-Markov substitution models, an approach of establishing a common ground for a fair and consistent comparison between them is by converting the non-Markov models into Markov models. Not many have attempted to convert a non-Markov amino acid substitution model into a Markov substitution model. A notable effort is made by Veerassamy et al. (2003) who presented an approximation of substitution probabilities in the BLOSUM series as a function of PAM evolutionary time. Their objective was to arrive at a Markov model of evolution which is compatible with BLOSUM.

In this thesis, any given non-Markov substitution matrix is converted into a Markov form by following the simple properties of Markov models discussed in the previous section. This facilitates a consistent comparative analysis between existing substitution models by bringing all those matrices into the context of a DTMC. Below lists the steps followed for this conversion.

Steps for Conversion

In the context of this thesis, all non-Markov models go through the following steps 1-3.

1. Convert a published, log odds scoring matrix into its conditional probability form using the Equation 6.1. (Note: In case the original amino acid frequencies are absent, use a null probability model which is optimal under the experimental setting of this thesis, in order to compute the denominator (i.e. the unrelated probability term) of the log odds ratio in the equation – in that way, it will be treated in the fairest way possible).

2. If the derived substitution probability matrix does not reflect a 1% expected change, assume it as representing some $\mathbf{M}(t)$ of a Markov model and derive its approximate base matrix $\tilde{\mathbf{M}}(1)$ by finding the k^{th} matrix root which is the closest to 1% expected change.

3. Scale the resultant base matrix with a current expected change `curr` to possess a 1% expected change, under the assumption that evolutionary time t is linear to the expected change within a small interval $t \pm \delta t$.

$$\mathbf{M}(1) = \tilde{\mathbf{M}}(1)^{\frac{0.01}{\texttt{curr}}}$$

Converting a non-Markov model into a Markov model through the aforementioned approach enables its application under the MML protein alignment framework introduced in this thesis. Since a pairwise alignment is always generated using the substitution matrix suited optimally for the evolutionary time between the two proteins in comparison, the original non-Markov matrix will anyway be applied if the pair depicts an expected change reflected by that particular matrix.

6.2.5 Converting a Rate Matrix into a Markov Matrix

For a given mixture of Markov and non-Markov models, the above section grounded non-Markov models in a DTMC context. Independently, for Markov models, if a substitution scoring matrix is originally from a DTMC, only step 1 of the above listed conversion steps is required. On the other hand, if the originally published version is a CTMC, the below steps are performed to obtain the base conditional probability matrix that accounts for 1% expected change.

1. If an exchangeability matrix $\mathbf{E}$ has been published, retrieve the corresponding instantaneous rate matrix $\mathbf{Q}$ with the published amino acid frequencies as the $\vec{\pi}$ stationary distribution, using the relationship defined in Equation 6.5 (Note: In case the original amino acid frequencies are absent, use a null probability model which is optimal under the experimental setting of this thesis)

2. If the originally published or the above derived rate matrix does not reflect a 1% expected change, normalise the matrix by scaling each cell $\mathbf{Q}(i,j)$ as:

$$\mathbf{Q}(i,j) = \frac{\mathbf{Q}(i,j)}{\texttt{curr} \times 100}$$

3. Convert $\mathbf{Q}$ to $\mathbf{M}(1)$, by defining $t = 1$ via the Equation 6.4

Now that we have established the essential background on Markov models of amino acid substitution, let us next go into the details on the essence of this chapter. The objective is to infer an optimal Markov matrix given a benchmark set of protein alignments. The next sections present an inference method followed by a thorough comparative analysis and discussion on the properties of a selection of existing substitution models and a set of newly inferred Markov models over several benchmarks.

6.3 MML based Substitution Matrix Inference

This section describes an MML based MCMC inference procedure to attain an optimal Markov matrix of amino acid substitution over any given benchmark of protein alignments, taking both matched and gapped regions between sequences into account.

6.3.1 Formulating the Problem in the MML Framework

Let $\mathbf{D}$ denote any benchmark dataset of aligned protein sequences. Formally, it is composed of pairs of amino acid sequences and their *given* alignments:

$$\mathbf{D} = \{\langle \mathcal{A}_1, \mathbf{S}_1, \mathbf{T}_1 \rangle, \langle \mathcal{A}_2, \mathbf{S}_2, \mathbf{T}_2 \rangle, \ldots, \langle \mathcal{A}_{|\mathbf{D}|}, \mathbf{S}_{|\mathbf{D}|}, \mathbf{T}_{|\mathbf{D}|} \rangle\}$$

where each protein sequence pair $\langle \mathbf{S}_i, \mathbf{T}_i \rangle$ is assigned an alignment relationship $\mathcal{A}_i$ specified as a three-state string over `match`(`m`), `insert`(`i`) and `delete`(`d`) states (Recall the three-state machine illustrated in Figure 2.4 which generates such an alignment string). The objective is to encode this observed dataset $\mathbf{D}$ losslessly and efficiently. Below lists the key statistical models required for this purpose.

A Markov matrix M: This is an amino acid substitution matrix which is used to losslessly encode the corresponding pairs of amino acids in each sequence pair $\langle \mathbf{S}_i, \mathbf{T}_i \rangle$ that are under `m` states in $\mathcal{A}_i$. Recall §4.1.1 that discusses this encoding in detail. Here, we apply the stationary distribution of $\mathbf{M}$ as the *null model* probabilities of amino acids, such that the joint probability of an amino acid pair is computed as per the detailed balance in Equation 6.2. (Note: $\mathbf{M}$ can be either an existing substitution matrix prepared using the steps listed in §6.2.4 and §6.2.5, or optimally inferred under the MML criterion as explained in §6.3.2).

A multistate model P: This is a 20-state model which gives *null model* probabilities of the 20 amino acids. It is used to losslessly encode the amino acids in the unaligned regions of $\langle \mathbf{S}_i, \mathbf{T}_i \rangle$ (i.e. under `i` and `d` states). Recall §4.4 on how to infer such *null model* estimates over a collection of protein sequences. In this chapter, the estimates of $\mathbf{P}$ are optimally inferred from amino acid subsequences that come under the insertion and deletion regions of the alignments in $\mathbf{D}$. Two other possible choices for $\mathbf{P}$ are: (1) the stationary distribution of the stochastic matrix $\mathbf{M}$, and (2) the estimates derived from a source independent of $\mathbf{D}$ (such as the UniProt database (UniProt Consortium and others, 2017)).

The set of evolutionary time parameters $\boldsymbol{\tau}$: This contains the optimal evolutionary time parameters $\boldsymbol{\tau} = \{t_1, t_2, ..., t_{|\mathbf{D}|}\}$, inferred for each sequence-pair in $\mathbf{D}$. As already established in this thesis during multiple instances, any t_i captures the evolutionary divergence between the corresponding sequence-pair $\langle \mathbf{S}_i, \mathbf{T}_i \rangle$, interpreted as the number of steps of the Markov chain by which their amino acids are related under the given Markov model $\mathbf{M}$ of amino acid substitutions. Recall §4.1.1 and §4.1.2 that explain how this optimal time parameter is searched for a given sequence alignment. The method applies a modified variant of the bisection search over the integral values of $t \in [1, t_{\max} = 1000]$, where in each iteration, the interval is reduced by half (i.e. the search range is halved). The variation attempts to handle some special cases to avoid the method from being trapped within in a local minimum. Here t_{max} was chosen to be 1000 because, as pointed out in §6.2, a Markov matrix $\mathbf{M}(t)$ reaches its stationary distribution when $t \to \infty$. Examining the convergence plots of the substitution matrices (see §6.5), $t = 1000$ seems to be in the range where any substitution matrix loses its differentiation ability. Thus it is a practical choice.

A set of Dirichlet parameters $\boldsymbol{\alpha}$: This contains parameter vectors for 1-simplex and 2-simplex Dirichlet distributions that explain the three free parameters ($\Pr(\mathtt{m}|\mathtt{m})$, $\Pr(\mathtt{i}|\mathtt{i})$ and $\Pr(\mathtt{m}|\mathtt{i})$ as highlighted in Figure 4.1) of the three-state machine as a function of evolutionary time t. All mathematical details, estimation methods and insights related to these models are detailed in Chapter 5. Each "evolutionary time" t value in the range $[1, t_{\max} = 1000]$ maps with a distinct 1-simplex Dirichlet model and 2-simplex Dirichlet model. These Dirichlet models are priors for inferring the three-state machine parameters of each alignment string in $\mathbf{D}$ using the Equation 3.25 under the MML87 method.

The set of three-state transition probabilities $\boldsymbol{\Theta}$: This contains a vector $\vec{\Theta}_i$ comprising the three free parameters $\Pr(\mathtt{m}|\mathtt{m})$, $\Pr(\mathtt{i}|\mathtt{i})$ and $\Pr(\mathtt{m}|\mathtt{i})$ for any alignment string $\mathcal{A}_i$ in $\mathbf{D}$. They have been inferred using one of the suitable time-dependent (1-simplex and 2-simplex) Dirichlet priors mentioned above, and are encoded using the same. They are in turn applied to losslessly encode the three-state alignment string $\mathcal{A}_i$.

The objective function

The task here is to compute the total encoding length of the message required for communicating the benchmark $\mathbf{D}$ of protein alignments using the above described models. Adhering to the MML Equation 3.10 of estimating the Shannon information content required for encoding some observed data D and their hypotheses H jointly, we can define the current message length formulation by the following summation of individual Shannon information terms:

$$I(\mathbf{H},\mathbf{D}) = I(\mathbf{M}) + I(\mathbf{P}) + I(\boldsymbol{\alpha}) + \sum_{i=1}^{|\mathbf{D}|} I(t_i) + I(\vec{\Theta}_i|\boldsymbol{\alpha},t_i) + I(\mathcal{A}_i|\vec{\Theta}_i,t_i) + I(\langle \mathbf{S}_i,\mathbf{T}_i\rangle|\mathcal{A}_i,\mathbf{M},\mathbf{P},t_i) \tag{6.6}$$

where,

$\mathbf{H}$:	accounts for all model parameters used to explain $\mathbf{D}$;
$I(\mathbf{M})$:	is the lossless encoding length of the base Markov matrix $\mathbf{M}^1$ related to evolutionary time $t = 1$ (with 1% expected change);
$I(\mathbf{P})$:	is the statement length of the probability model $\mathbf{P}$ which gives *null model* estimates of amino acids;
$I(\boldsymbol{\alpha})$:	is the statement length of the set of time-dependent Dirichlet parameters $\boldsymbol{\alpha}$;
$I(t_i)$:	is the statement length of the optimal evolutionary time t_i inferred for a sequence-pair $\langle \mathbf{S}_i,\mathbf{T}_i\rangle$ given its alignment $\mathcal{A}_i$;
$I(\vec{\Theta}_i\|\boldsymbol{\alpha},t_i)$:	is the statement length of the free parameter vector $\vec{\Theta}_i$ of the three-state alignment machine inferred for each $\mathcal{A}_i$;
$I(\mathcal{A}_i\|\vec{\Theta}_i,t_i)$:	is the statement length of each three-state alignment string $\mathcal{A}_i$;
$I(\langle \mathbf{S}_i,\mathbf{T}_i\rangle\|\mathcal{A}_i,\mathbf{M},\mathbf{P},t_i)$:	is the statement length of explaining all amino acids in each sequence-pair $\langle \mathbf{S}_i,\mathbf{T}_i\rangle$;

Computing the Shannon information terms of $I(\boldsymbol{\alpha})$, $I(t_i)$, $I(\vec{\Theta}_i|\boldsymbol{\alpha},t_i)$, $I(\mathcal{A}_i|\vec{\Theta}_i,t_i)$ and $I(\langle \mathbf{S}_i,\mathbf{T}_i\rangle|\mathcal{A}_i,\mathbf{M},\mathbf{P},t_i)$ appearing in the above objective function were detailed in the previous Chapters 4 and 5. Following deals with the computation of $I(\mathbf{M})$ and $I(\mathbf{P})$ terms.

$I(\mathbf{M})$: This is the message length of communicating the base stochastic matrix $\mathbf{M}(1)$ of the Markov model as a set of twenty points in a unit 19-simplex (representing the 20 column vectors). Each column vector $\vec{v_j}$ is encoded by assuming a uniform prior $h(\vec{v_j}) = \frac{19!}{\sqrt{20}}$. Accordingly, the MML formulation of $\vec{v_j}$ (as per the Equation 3.10) is given by:

$$I(\vec{v_j}) = \frac{19}{2}\log(c_{19}) - \log(h(\vec{v_j})) + \frac{1}{2}\log\left(\frac{N_j^{19}}{\prod_{\forall p\in\vec{v_j}} p}\right) \quad \text{bits}$$

where N_j is the total count of amino acid transitions represented by $\vec{v_j}$ in the benchmark dataset, and c_{19} is the quantising lattice constant (Conway and Sloane, 1984) associated with

19 free dimensions. Summing up the message lengths of all the 20 column vectors, we get:

$$I(\mathbf{M}) = \sum_{j=1}^{20} I(\vec{v_j}) \quad \text{bits}$$

$I(\mathbf{P})$**:** This is the message length of communicating the *null model* probabilities of amino acids as a 20-state distribution for encoding amino acid insertions and deletions. It is again a point in a unit 19-simplex. Thus, similar to the previous case of $\vec{v_j}$ above, the MML formulation of $\mathbf{P}$ under a uniform prior (as per the Equation 3.10) is given by:

$$I(\mathbf{P}) = \frac{19}{2}\log\left(c_{19}\right) - \log\left(h(\vec{v_j})\right) + \frac{1}{2}\log\left(\frac{X^{19}}{\prod_{\forall p \in \mathbf{P}} p}\right) \quad \text{bits}$$

where X is the total number of insertions and deletions in the benchmark dataset, and c_{19} is the quantising lattice constant (Conway and Sloane, 1984).

6.3.2 Search for the Best Markov Matrix of Amino Acid Substitution

Given the earlier defined MML objective function (Equation 6.6), we can now compute the total encoding length of the message that communicates the complete protein alignment benchmark $\mathbf{D}$. In other words, it gives the Shannon information content of $\mathbf{D}$ under the statistical models/parameters: $\{\mathbf{M}, \mathbf{P}, \boldsymbol{\alpha}, \boldsymbol{\tau}, \boldsymbol{\Theta}\}$. The primary goal is to optimise the Markov matrix $\mathbf{M}$ of amino acid substitutions under this objective function. Since $\boldsymbol{\alpha}$ Dirichlet parameters are directly connected to $\mathbf{M}$ via the evolutionary time parameter t, we can employ an Expectation-Maximisation (EM) like strategy for the simultaneous optimisation of $\mathbf{M}$ and $\boldsymbol{\alpha}$.

The EM-like search involves holding the $\boldsymbol{\alpha}$ fixed (and consequently, the $\boldsymbol{\Theta}$ fixed), while performing a Markov Chain Monte Carlo (MCMC) search for the best, optimal matrix $\mathbf{M}$ and its associated $\boldsymbol{\tau}$. Afterwards, $\mathbf{M}$ and $\boldsymbol{\tau}$ are held fixed to estimate the optimal $\boldsymbol{\alpha}$ (and consequently, the $\boldsymbol{\Theta}$). This process is repeated until the objective function converges.

Search for the Best Stochastic Matrix M with Fixed $\{\boldsymbol{\alpha}, \boldsymbol{\Theta}\}$

While holding the Dirichlet and three-state machine parameters ($\boldsymbol{\alpha}$ and $\boldsymbol{\Theta}$) fixed, the current state of the stochastic (base $t = 1$) matrix is optimised over all set of sequence-pairs and their alignments in the benchmark $\mathbf{D}$ using a Simulated Annealing (SA) approach (See §5.3.4 for the preliminary discussion on SA).

Initially, the search starts with a reasonable stochastic matrix for $\mathbf{M}$, and lets it undergo an SA process. As noted by Qian and Goldstein (2002), the space of all possible substitution matrices is complex, and thus, there is an inevitable chance that any inferred matrix is a local optimum based on the starting point of the search. A good starting point would be an existing and timely substitution matrix which has been prepared as a stochastic base ($t = 1$) matrix by following the steps listed in §6.2.4. Similarly, $\boldsymbol{\alpha}$ inferred from any of the earlier matrices is a good starting point. Hence, for all practical purposes, it was decided to begin the optimisation process with a Markov matrix prepared from the latest PFASUM-60 scoring matrix (which comes under the non-Markov matrix series of PFASUM (Keul et al., 2017) – see §6.4.1 for details). On the other hand, the initially fixed $\boldsymbol{\alpha}$ is the 1-simplex and 2-simplex Dirichlet models inferred as a function of PAM-t evolutionary time in Chapter 5.

Simulated Annealing Setup: Starting from the above matrix as the initial system state under an initial temperature parameter of $T = 10,000$, we gradually cool down the system using a cooling schedule. There, the temperature is decreased by a factor of 0.88 after 500 perturbations of the stochastic matrix (i.e. evolving a Markov chain of system states) at each temperature mark. Each new Markov chain (that corresponds to a certain temperature) starts with the best matrix achieved so far. The process continues until a temperature $T = 0.0001$.

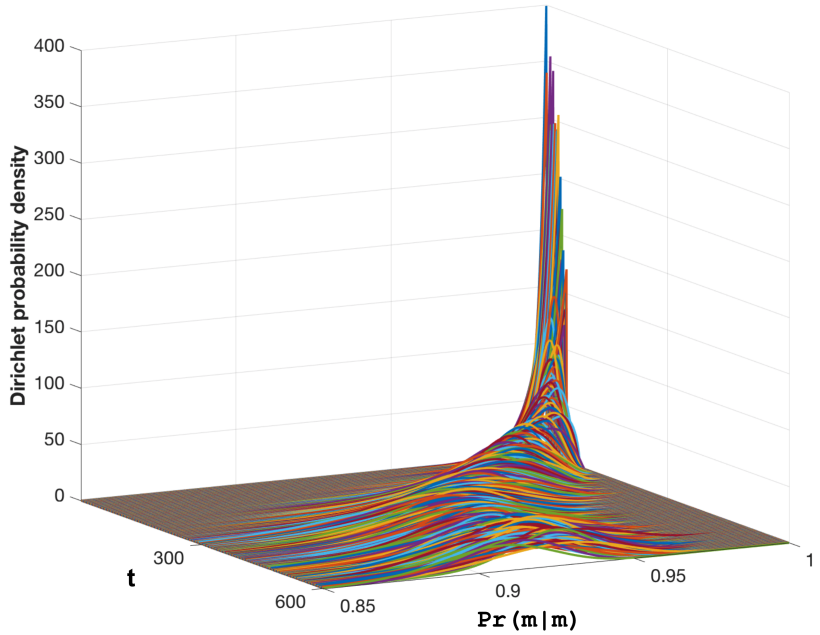
Matrix Perturbation: At each iteration, the current matrix $\mathbf{M}$ is perturbed by randomly selecting one of its column vectors to perturb. This random selection is made according to the stationary distribution of the matrix. The selected column vector is perturbed by sampling a new vector from a Dirichlet distribution $\text{Dir}(\vec{\alpha} = \vec{\mu}\kappa)$, where $\vec{\mu}$ is the selected column and κ is a specified high concentration parameter κ (Note the reparameterisation given in §5.2.1). This defines a 20-dimensional Dirichlet distribution explaining a unit 19-simplex. (Sampling a vector from a Dirichlet distribution was previously described in §5.2.1 with a pseudocode given in Figure 5.4). Initially, κ is set to $1,000,000$. As the system cools down by a factor of 0.88 at each temperature mark, the Dirichlet sampling distribution of a column vector is made even more concentrated by increasing the κ by a factor of $\frac{1}{0.88}$. This makes the neighbourhood of sampling tighter and tighter as the SA process continues and the matrix $\mathbf{M}$ converges. During this perturbation step, the validity of the perturbed matrix is always ensured (i.e. the matrix is properly normalised and has real eigenvalues).

For each perturbation of $\mathbf{M} \rightarrow \tilde{\mathbf{M}}$, the total lossless encoding length as per the Equation 6.6 is recomputed using $\tilde{\mathbf{M}}$. The new matrix is accepted or rejected based on the Metropolis criterion. If the encoding length decreases, then the change is accepted with a probability of 1; Otherwise, it is accepted with a probability of $2^{-\frac{\Delta I}{T}}$, where ΔI is the difference in the encoding lengths under $\tilde{\mathbf{M}}$ versus $\mathbf{M}$.

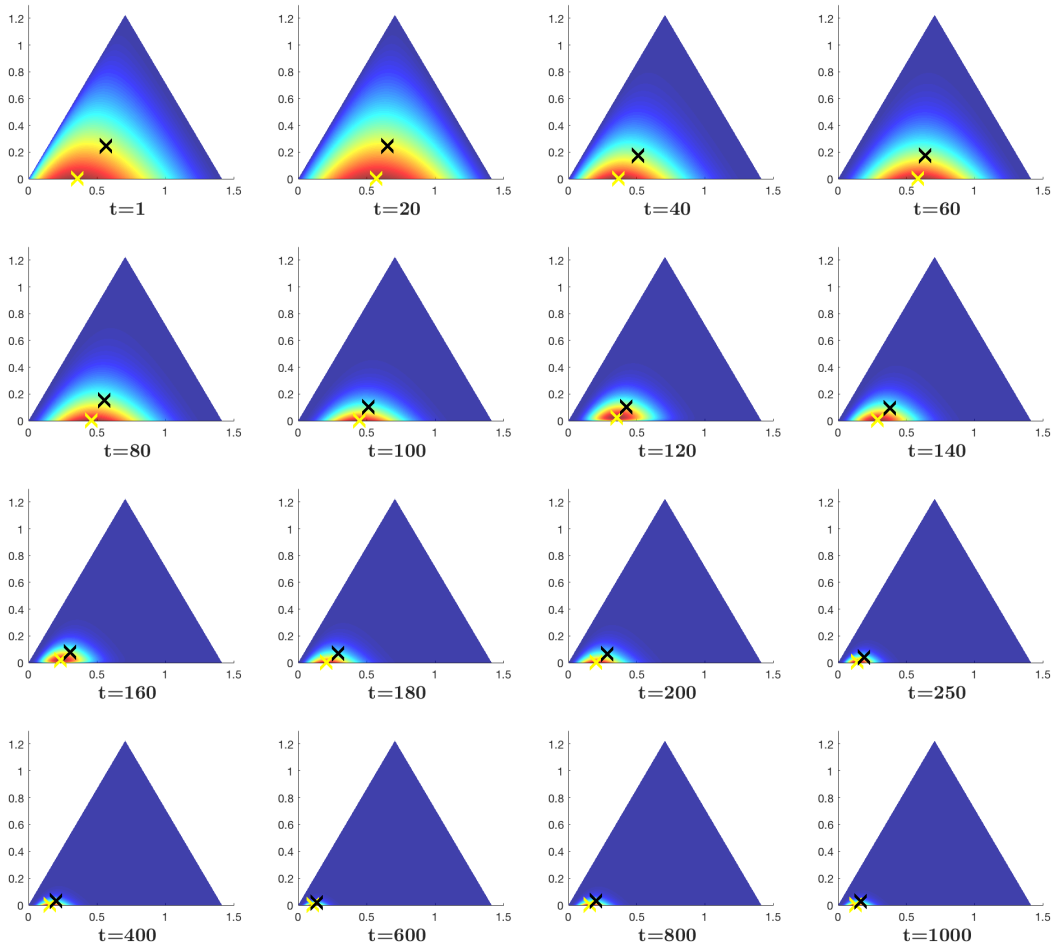
Search for the Time-dependent Dirichlet Parameters $\boldsymbol{\alpha}$ with Fixed $\{\mathbf{M}, \boldsymbol{\tau}\}$

Holding the stochastic matrix $\mathbf{M}$ and time parameters $\boldsymbol{\tau}$ fixed, we can group all the alignments in the benchmark $\mathbf{D}$ into their respective discrete bins of evolutionary time. This results in subsets of alignments $\mathbf{A}(t)$ for each $t \in [1, t_{\max} = 1000]$. The goal is to now estimate 1-simplex and 2-simplex Dirichlet models for each discrete time-bin over the respective three-state alignment strings. Chapter 5 introduced the MML inference method for estimating these models with an MCMC search for optimal Dirichlet parameters in §5.3.4. Accordingly, α is re-estimated to be used in conjunction with the best substitution matrix $\mathbf{M}$ inferred in the previous step. Then, a new $\boldsymbol{\Theta}$ set is estimated under the updated $\boldsymbol{\tau}$ as per the Equation 3.25.

The above detailed optimisation method arrived at an improved Markov matrix of amino acid substitutions and its associated evolutionary-time-parameterised Dirichlet models over a structural alignment benchmark of 59,092 SCOP protein domains pairs. The new substitution model is named as the **M**inimum **M**essage **L**ength **SU**bstitution **M**atrix (**MMLSUM**). The MML protein alignment framework introduced in Chapter 4 can now benefit from these improved models of amino acid substitution and three-state alignment strings as a function of evolutionary time. Figure 6.2 illustrates how the respective 1-simplex and 2-simplex Dirichlet probability distributions vary as a function of evolutionary time t under the MMLSUM Markov model. Furthermore, five other structural alignment benchmarks were employed to obtain five more Markov substitution models, making the total number of inferred matrices as six including the MMLSUM matrix.



(a) 1-simplex Beta plots



(b) 2-simplex Dirichlet plots

Figure 6.2: Visualisation of the inferred Dirichlet distributions, modelling the three free parameters of the finite state machine. (a) 1-simplex distributions of Pr(m|m) associated with state m, as a function of evolutionary time $t \in [1, 600]$, and (b) 2-simplex distributions of Pr(i|i) and Pr(m|i) associated with state i (and by symmetry, state d), as a function of evolutionary time $t = \{1, 20, 40, 60, 80, 100, 120, 140, 160, 180, 200, 250, 400, 600, 800, 1000\}$, under the new MML-SUM substitution model. The yellow cross marker and black cross marker highlight the mode and mean vectors, respectively.

The next section conducts a comparative analysis of a collection of nine existing substitution matrices and those aforementioned six MML matrices. MMLSUM has the best overall performance amongst all the matrices in comparison as shown in §6.4.3. An extended analysis of this matrix reveals further insights regarding amino acid properties, protein function and evolution in §6.5 and §6.5.4.

6.4 Comparative Analysis, Results and Insights

Here, this thesis presents a comparative study across nine existing substitution models, as well as a new set of Markov models inferred over six different structural alignment benchmarks (following the inference method described in §6.3). The existing matrices covered widely-used, as well as recently published substitution models (both Markov and non-Markov) since the pioneering work of Dayhoff et al. (1978). To conduct an impartial and consistent comparison, all non-Markov matrices were brought down to their base form of 1% expected change, assuming their original form as reflecting a discrete time point in a Markov model, as explained in §6.2.4.

The below section lists and details the existing matrices used in this study, following the next section giving information about the alignment benchmarks that were employed to infer new Markov models.

6.4.1 Selection of Matrices

This study looked at the following amino acid substitution models:

PAM (Dayhoff et al., 1978): This is the oldest Markov model proposed to capture amino acid evolution, which in turn inspired many subsequent improvements. It is still a widely-used matrix (particularly the PAM-250 scoring matrix) within many distributed sequence alignment programs. As described with more details in §6.1.2, it was derived by Dayhoff et al. (1978) over a small and limited dataset, covering only 1572 substitutions (with no observed replacements for some amino acid pairs). This thesis computes the PAM-1 conditional probability matrix using raw data published in the original paper (i.e. the substitution count matrix, relative mutabilities and relative frequencies).

JTT (Jones et al., 1992a): JTT (also known as PET91) is an improved version of PAM. The main difference lies in the use of an approximate method for phylogenetic tree inference instead of the maximum parsimony method used by Dayhoff et al. (1978). The authors identified potential homology between two proteins by analysing their 3-mer frequencies (i.e. a higher number of identical 3-mers suggests greater homology), via a normalised 3-mer frequency score denoting the fractional area of overlap between two 3-mer histograms. This is regarded as a *heuristic measure of identity*. They also examined how this measure varies with Needleman-Wunch alignment scores (under a constant gap) and the resultant percent identities. Next, they constructed a distance matrix based only on sequence pairs with $> 85\%$ identity, and performed single-linkage clustering. Substitution statistics were derived in terms of each sequence based on its alignment with another sequence that gives the highest pairwise alignment score. There are two versions of the PET91 conditional probability matrix (equivalent to PAM1): one based on PIR release 29, and the other one based on Swiss-Prot Release 22, available at `http://bioinfadmin.cs.ucl.ac.uk/downloads/Matrices`. This thesis uses the latter version.

BLOSUM (Henikoff and Henikoff, 1992): The BLOSUM series acts as a landmark, non-Markov model for substitution modelling. The method of deriving this series was discussed in §6.1.3. It is the most used non-Markov matrix series, with BLOSUM-62 being the common choice for aligning averagely distant pairs.[2] BLOSUM-45 and BLOSUM-90 are common choices for more distant and closely related pairs, respectively. Theoretically, any BLOSUM-n matrix can be obtained over a block of ungapped, multiple alignments. The authors originally published 16 target frequency (joint probability) matrices and scoring matrices (at `ftp://ftp.ncbi.nih.gov/repository/blocks/unix/blosum` – latest version 5.0 in 1992) for $n \in \{30, 35, \ldots, 90, 95, 100\}$ and $n = 62$. They have been inferred over 2000 ungapped multiple alignment blocks from the BLOCKS database (Pietrokovski et al., 1996), across more than 500 groups of related proteins, with at least 2369 changes for each possible amino acid pair. The conditional probability distributions are directly obtainable from those joint probability matrices, as the null probability distribution of amino acids is reflected by the marginal distribution. Note: This thesis takes BLOSUM-62 as the representative matrix of the model.

Johnson-Overington matrix (Johnson and Overington, 1993): Denoted by JO, this is an early, non-Markov matrix that was estimated on structural alignment data. The originally published substitution count matrix can be used to derive a conditional probability matrix with maximum likelihood estimates over the count data (i.e. $\Pr(a_i|a_j) = n_{a_i,a_j} / \sum_{\forall a_i} n_{a_i,a_j}$ for every amino acid pair a_i and a_j, as defined in the original paper). Their dataset covered 207,795 amino acid exchanges present in 65 homologous sets of 3D structural alignments across 235 proteins, mainly incorporating 15% to 40% sequence identity.

WAG (Whelan and Goldman, 2001): This matrix mainly represents substitution propensities in globular protein family sequences, inferred as a CTMC over 3905 proteins across 182 families from an earlier existed and unpublished sequence alignment database called BRKALN. Whelan and Goldman (2001) have followed an EM based Maximum-Likelihood method to estimate a count based rate matrix $\mathbf{Q}$, by defining the likelihood of the model given the families of aligned proteins and their phylogenetic trees (with relative branch lengths), and iteratively re-inferring $\mathbf{Q}$ and the phylogenetic relationships until convergence. The WAG rate matrix has been published with its associated amino acid frequencies at `https://www.ebi.ac.uk/goldman-srv/WAG/wag.dat`. Le and Gascuel (2008) note a potential bias in the used dataset in terms of globular proteins that are easy to be crystallised (due to the fact that, a protein family has been included if at least one protein in the family had a resolved 3D structure at the time BRKALN was developed in the mid 90s).

VTML (Müller et al., 2002): VTML stands for Variable Time Maximum Likelihood, presented as a CTMC model. Originally, Müller and Vingron (2000) proposed the VT matrix, inferred via a matrix resolvent method to iteratively estimate the rate matrix over pairwise sequence alignments from the SYSTERS protein family database (Krause et al., 2000) (over 2.7 million amino acid pairs). They have used PAM as the starting point for the search. This method have also accounted for the optimal evolutionary divergence between proteins. Later, VTML was inferred using an iterative ML estimation. The Perl script for generating the VTML1 matrix is available at: `https://owww.molgen.mpg.de/~muelle_t/vt_scores`

[2]Both BLOSUM and PAM appear as substitution matrix choices for the BLAST (Altschul et al., 1990) alignment program.

LG (Le and Gascuel, 2008): This is an improved version of the WAG (CTMC) model, covering more families and accounting for rate variations across amino acid sites (through a set of gamma distributed rate categories). This incorporated 3912 seed (multiple) sequence alignments from the Pfam database (Bateman et al., 2000), over around 50,000 sequences with approximately 6.5 million amino acids. Overall, 3912 alignments with a limited number of gaps ($\sim$ 1% of the amino acids being inserted/deleted) were involved. They also followed an EM based ML estimation similar to the procedure of WAG inference. The rate matrix and the associated amino acid frequencies are available from: http://www.atgc-montpellier.fr/download/datasets/models/lg_LG.PAML.txt.

MIQS (Yamada and Tomii, 2013): MIQS was derived through a PCA analysis of several existing scoring matrices including BLOSUM and VTML, by sampling a new point from the PCA subspace defined by the first three principal components. Since there is no specific null amino acid distribution involved, this thesis applied the optimal *null model* MML probability estimates of amino acids over the 59,092 benchmark structural alignment dataset (described in §5.3.1), when deriving the associated conditional probability matrix as per §6.2.4. This ensures fair treatment. The MIQS scoring matrix (published in 3 bit units) was retrieved through the DECIPHER Bioconductor package (https://rdrr.io/bioc/DECIPHER/man/MIQS.html). A scaling factor $c = 3$ was applied during the conversion of this scoring matrix to its conditional probability form using the Equation 6.1.

PFASUM (Keul et al., 2017): This is the most recent, BLOSUM-like family of matrices, derived over Pfam seed multiple structural alignments. Unlike BLOSUM, PFASUM has taken gapped alignments into account, while also considering special amino acids. The scoring matrix series is available at: http://www.cbs.tu-darmstadt.de/PFASUM/. As the corresponding amino acid frequencies have not been published, this thesis computed a conditional probability matrix of any PFASUM-n scoring matrix in the same way as done for the MIQS matrix. It should also be noted that, PFASUM-n scoring matrices for the ranges of $30 \leq n \leq 42$, $43 \leq n \leq 73$ and $74 \leq n \leq 100$ have been published by the authors in 4 bit units, 3 bit units and 2 bit units, respectively. Thus, the appropriate conversion was applied to 16 matrices covering $n \in \{30, 35, ..., 90, 95, 100\}$ and $n = 62$ matrices (equivalently comparable to the BLOSUM series). Scaling factors were assigned suitably (i.e. $c = 4$,$c = 3$ and $c = 2$, respectively for Equation 6.1). This thesis takes PFASUM-60 as the representative matrix of the model, since the authors have recommended it to be a general choice for distant relationship detection based on their evaluations.

Altogether, this selection of historical and recent matrices consists of five Markov models and four non-Markov models, representing the four decades long timeline of efforts for amino acid substitution modelling (from 1978 to 2020).

6.4.2 Selection of Alignment Benchmarks

This thesis employed the following structural alignment benchmarks to validate the MML matrix optimisation framework introduced in §6.3. Each benchmark individually provides a source collection $\mathbf{D}$ of protein sequence pairs and their alignment relationships. (Note: all possible pairwise relationships were extracted from multiple alignment benchmarks.) The primary basis for comparing substitution matrices is the Shannon information content of encoding all protein

sequences in a given benchmark. Since a substitution model closest to the true model is expected to encode these aligned protein sequence data as economically as possible, we expect the best model to give the lowest amount of information (ie. the shortest message length). In information theoretic terms, a better substitution model improves compression of those sequences by effectively utilising their pairwise structural relationships. The **D** is losslessly compressed under the MML criterion (given by Equation 6.6).

This study took the following benchmarks of structural alignments:

HOMSTRAD (Mizuguchi et al., 1998) (https://mizuguchilab.org/homstrad) is a database of multiple structural alignments for homologous protein families. It covers 1032 families with known structures. Their alignments are semi-manually curated using the structural alignment programs: MNYFIT, STAMP and COMPARER (Sutcliffe et al., 1987; Russell and Barton, 1992; Sali and Blundell, 1990).

Mattbench (Daniels et al., 2011) (https://bcb.cs.tufts.edu/mattbench/Mattbench.html – version 1.0) database has been curated using the structural alignment program MATT (Menke et al., 2008). This thesis combines Mattbench's two sets of alignments classified as *superfamily* and *twilight zone* into a single benchmark. The superfamily set contains alignments of 225 groups of homologous protein domains, where all pairs of domains in any group have a sequence identity $< 50\%$. The twilight zone set is a much smaller and distinct set, containing alignments covering 34 distantly related groups where the sequence identity threshold is $< 20\%$ (Daniels et al., 2011).

SABMARK (Van Walle et al., 2005) (http://bioinformatics.vub.ac.be/databases/databases.html – version 1.65) is a more extensive set of alignments covering *superfamily* and *twilight zone* sets of protein domains, whose alignments have been generated using SOFI and CE (Boutonnet et al., 1995; Shindyalov and Bourne, 1998). The superfamily set (`SABMARK-sup`) contains 425 groups of multiple alignments, while the twilight zone set (`SABMARK-twi`) contains 209 groups.

SCOP (Andreeva et al., 2020) (https://scop.berkeley.edu) database (v2.07) was used to derive a set of 59,092 unique protein domain pairs, randomly sampled from the *superfamily* (36,372) and *family* (22,720) levels of its hierarchy. (This source collection was resultant from the sampling and filtering procedure described in §5.3.1). These 59,092 pairs were aligned separately using DALI (Holm and Sander, 1993) (`DaliLite.v5` from http://ekhidna2.biocenter.helsinki.fi/dali/) and MMLigner (Collier et al., 2017) (https://lcb.infotech.monash.edu/mmligner) structural alignment programs to provide two alignment benchmarks: `SCOP1` and `SCOP2`. `SCOP1` has only $56,292$ structural alignments, as DALI did not produce any alignment for 2800 pairs.

The descriptive statistics of all involved benchmarks are listed in Table 6.1.[3] `SCOP2` can be taken as the most representative benchmark since it is the largest in size (i.e total number of pairwise alignments), covering a good distribution of sequence identity percentages indicating

[3]Sequence identity percentage of a pair of proteins is computed as the number of matched, identical amino acid pairs between two sequences divided by the length of the shorter sequence.

Table 6.1: Structural alignment benchmark datasets (with their total number of aligned pairs (size), total number of matches, total number of insertions, total number of deletions, and the mean sequence identity percentage)

Dataset	Size	# of ms	# of is	# of ds	Mean id. %
HOMSTRAD	8323	1,311,478	96,911	98,810	35.1
Mattbench	5286	826,506	177,401	177,789	19.4
SABMARK-Sup	19,092	1,750,440	848,859	861,344	15.2
SABMARK-Twi	10,667	694,954	515,318	527,188	8.4
SCOP1	56,292	8,663,652	1,407,988	1,373,882	25.5
SCOP2	59,092	8,145,678	1,673,687	1,653,531	24.8

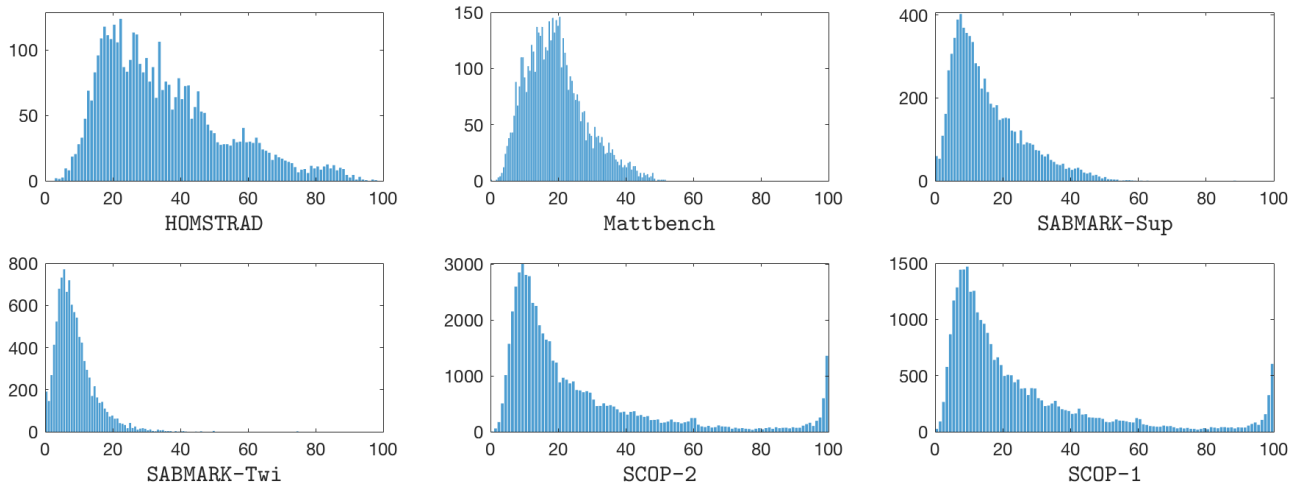


Figure 6.3: Sequence identity percentage distributions (histograms) across the six structural alignment benchmarks

varying degrees of evolutionary divergence.[4] See Figure 6.3 for sequence identity percentage histograms of each benchmark. As indicative from the histograms, Mattbench and SABMARK are more biased towards distant relationships. On the other hand, HOMSTRAD covers a wide range of evolutionary time.

An MML stochastic matrix was inferred over each of the above six benchmarks, resulting in six new matrices: $\text{MML}_{\texttt{HOMSTRAD}}$, $\text{MML}_{\texttt{MATTBENCH}}$, $\text{MML}_{\texttt{SABMARK-Sup}}$, $\text{MML}_{\texttt{SABMARK-Twi}}$, $\text{MML}_{\texttt{SCOP1}}$ and $\text{MML}_{\texttt{SCOP2}}$.[5] Their individual performance was compared to the previously listed existing substitution matrices as well as to each other, in Shannon information terms.

The purpose of incorporating several benchmarks was to test and validate the developed MML framework of Markov matrix optimisation for its general and consistent applicability as a method for optimising/evaluating the performance of any substitution matrix over any alignment benchmark. This selection of benchmarks is comprehensive since they represent different structural alignment hypotheses produced by a range of structural alignment programs.

6.4.3 Performance of Matrices across Benchmark Datasets

Here, the performance measure for evaluating any substitution matrix is the Shannon information content (defined by Equation 6.6) it incurs when encoding a given alignment benchmark.

[4] This is further justified by the performance of its inferred matrix across all the other benchmarks (highlighted in the next section §6.4.3).

[5] Later known as MMLSUM

All matrices were evaluated under three different choices (listed below), by varying the choice of **P** (i.e. the 20-state distribution that gives *null model* probabilities of amino acids) which is used to encode insertions and deletions.

Choice 1: The optimal 20-state *null model* probabilities estimated over the indel regions of the alignments in the respective benchmark using the MML87 method (§4.4)

Choice 2: The *stationary distribution* of the stochastic substitution matrix **M**

Choice 3: The optimal 20-state *null model* probabilities estimated over the database of UniProt protein sequences using the MML87 method (§4.4)

Both Choice 1 and 3 ensure a constant and independent *null model* application throughout the optimisation. Thus, we can exclude the information term $I(\mathbf{P})$ from the objective function (Equation 6.6) during matrix optimisation. In contrast, the objective function under choice 2 is controlled by a varying **P** at each iteration, as a perturbed matrix results in a perturbed stationary distribution. Consequently, two matrices (one under choice 1 and the other under choice 2) were inferred and evaluated over each benchmark. There is no need to infer another matrix under choice 3 since it is equivalent to choice 1. (Note: Only choice 1 requires **P** to be communicated. **P** under choice 2 can be obtained from the substitution matrix **M** itself. Further, choice 3 uses a universally applicable *null model* distribution which can be regarded as appearing in the *codebook*).

As with choices 1 and 3, the *null model* probabilities for matches and *null model* probabilities for indels do not necessarily have to be the same. (Recall that in this chapter, we use the stationary distribution of **M** as the *null model* probabilities for encoding matches, as described in §6.3). We can speculate that the evolutionary pressures these different regions go through may be different. As proteins evolve, they may freely accumulate indels, while matched regions undergo more pressure in conserving their amino acids.

Let us now evaluate the fifteen matrices (i.e. 9 existing matrices listed in §6.4.1 and the 6 newly inferred matrices listed in §6.4.2) under each of the above choices in terms of the Shannon information content required for encoding each benchmark of structural alignments.

Shannon Information Content of Benchmarks under Choice 1

Table 6.2 presents results for all matrices, where all parameters have been optimised for the respective benchmark. Their ranks of performance with respect to the particular benchmark are also listed beside their numerical figures (inside parenthesis). Here, the previously published matrices are arranged in their chronological order of publication. The last five rows show results for the stochastic matrices inferred from each benchmark.

Generally, a trend of model improvement can be observed for existing matrices with respect to their time of appearance in the field. To get an overall view of each matrix performance, a simple-yet-effective statistic is the (row-wise) sum of ranks of each matrix over all benchmarks, `ranksum` in short. This statistic acts as a consensus over all benchmarks generated from individual ranks. Since the evaluation here involves 15 matrices over 6 benchmarks, the `ranksum` of any matrix is an integer between $6 \times 1 = 6$ (i.e. the best possible performance) and $6 \times 15 = 90$ (i.e. the worst possible performance).

Amongst the set of existing matrices, `PAM` (`ranksum` = 90) consistently gave the worst (i.e. longest) lossless encoding lengths across all benchmarks. This is to be expected as `PAM` was derived in the late 70s using a limited set of closely related protein relationships available at that time. The noted shortcomings of its derivation in §6.1.2 explains this observation, affirming `PAM` as less representative of the current corpus of proteins. `PAM`'s performance is

Table 6.2: Shannon information content (in bits) to losslessly encode each benchmark by varying the substitution matrices under **choice 1**. The rank order of their performance is reported in each column within parentheses (smaller lossless encoding length = better rank). Best encoding length for each benchmark is highlighted. All other parameters are inferred (optimised) by the MML framework for the matrix chosen to losslessly compress each benchmark.

Matrix	HOMSTRAD	Mattbench	SABMARK-Sup	SABMARK-Twi	SCOP1	SCOP2
Shannon information using existing substitution models						
PAM (1978)	11531556.42 (15)	9143136.85 (15)	23574085.50 (15)	11310225.98 (15)	84925406.86 (15)	82757945.45 (15)
JTT (1992)	11481203.18 (14)	9072068.64 (13)	23450831.63 (13)	11251914.34 (13)	84353986.51 (13)	82218531.99 (13)
BLOSUM (1992)	11437552.82 (10)	9037049.84 (08)	23373908.14 (07)	11228043.62 (06)	84174710.81 (11)	81995179.31 (10)
JO (1993)	11476518.49 (13)	9118265.97 (14)	23501361.53 (14)	11290056.32 (14)	84567477.78 (14)	82405561.96 (14)
WAG (2001)	11419186.01 (05)	9052722.90 (12)	23400017.01 (11)	11243241.94 (12)	84141633.83 (09)	81996154.29 (11)
VTML (2002)	11423498.15 (07)	9035903.42 (07)	23377505.00 (08)	11230624.12 (08)	84075908.48 (07)	81925302.65 (07)
LG (2008)	11464263.57 (12)	9049040.59 (11)	23411713.27 (12)	11235389.21 (09)	84255656.90 (08)	82090289.00 (12)
MIQS (2013)	11422215.36 (06)	9040480.84 (10)	23385242.81 (10)	11236323.32 (10)	84076742.82 (12)	81927707.60 (08)
PFASUM (2017)	11412888.15 (02)	9039799.35 (09)	23379074.40 (09)	11236572.25 (11)	84040519.29 (04)	81902713.60 (06)
Shannon information using MML inferred substitution models						
$\text{MML}_{\text{HOMSTRAD}}$	**11405604.71 (01)**	9035317.57 (05)	23365151.19 (06)	11230184.94 (07)	84026302.25 (03)	81873575.93 (03)
$\text{MML}_{\text{MATTBENCH}}$	11426344.12 (09)	**9025882.40 (01)**	23355215.84 (03)	11217219.06 (03)	84050927.40 (05)	81886796.32 (04)
$\text{MML}_{\text{SABMARK-Sup}}$	11424135.90 (08)	9031315.87 (04)	**23346025.43 (01)**	11212252.65 (02)	84067892.89 (06)	81889152.29 (05)
$\text{MML}_{\text{SABMARK-Twi}}$	11442781.68 (11)	9035720.47 (06)	23356054.49 (05)	**11211360.90 (01)**	84155701.52 (10)	81962307.85 (09)
$\text{MML}_{\text{SCOP1}}$	11413295.62 (03)	9029682.81 (03)	23355235.85 (04)	11221295.64 (05)	**83996795.98 (01)**	81848381.03 (02)
$\text{MML}_{\text{SCOP2}}$	11413725.29 (04)	9028667.52 (02)	23349826.82 (02)	11218205.59 (04)	83999536.37 (02)	**81840654.64 (01)**

followed by the performance of JO (ranksum = 83), JTT (ranksum = 79), LG (ranksum = 64), WAG (ranksum = 60), MIQS (ranksum = 56), BLOSUM (ranksum = 52), VTML (ranksum = 44) and PFASUM (ranksum = 41). JTT indeed improved PAM. Despite being more recent and based on a more informative structural alignment dataset, JO lags behind JTT, yet with better compression than PAM. WAG and VTML arose roughly around the same time, yet VTML leads with a considerable difference. A counter-intuitive observation is that, LG is behind WAG in general performance according to ranksum, even though it was introduced as an improvement to WAG. However it performs better in three benchmarks: Mattbench, SABMARK-Twi and SCOP2. Further, the modern matrix MIQS lags behind BLOSUM and VTML models.

In general, the ranksums suggest that, by and large, the previously published models of amino acid substitutions have improved over time. However, BLOSUM is amongst the earliest matrices (published in 1992), which outperforms several other matrices that were proposed much later on. This substantiates BLOSUM's wide usage (particularly BLOSUM-62) in protein sequence alignment as an effective series of substitution matrices (See Table 6.6 for the series-wide performance of BLOSUM). It reflects the validity of the commonly accepted non-Markov BLOSUM series even in its non-trivial Markov context. However BLOSUM is superseded by VTML (published in 2002) and PFASUM (published recently in 2017). PFASUM being the recent BLOSUM-like matrix series, is again apparent to be performing the best amongst all the existing substitution models (See Table 6.6 for the series wide performance of PFASUM equivalent to BLOSUM). It is ahead of other existing models, explainable by its basis on the manually curated Pfam structural alignments.

In comparison, the inferred, benchmark-specific matrices perform consistently better than previously-published substitution matrices. This is precisely observed in Table 6.2 via the highlighted numerical figures. Such performance is expected, since the purpose of optimising those matrices (under the SA iterative search with PFASUM as the starting point – See §6.3.2) was to improve the current standing of amino acid substitution modelling. However, the utility of any matrix lies in its ability to generalise to *other* benchmarks and perform well on those. The Table 6.2 clearly demonstrates the ability of MML-inferred matrices to generalise and explain any benchmark, far outperforming all existing ones. From the point of view of their ranksums: $\text{MML}_{\text{SABMARK-Sup}}$ gives ranksum = 26 across all benchmarks, while $\text{MML}_{\text{MATTBENCH}}$ and $\text{MML}_{\text{HOMSTRAD}}$ give ranksum = 25. The top two performers overall come from the matrices inferred on the two SCOP benchmarks, $\text{MML}_{\text{SCOP1}}$ (ranksum = 18) and $\text{MML}_{\text{SCOP2}}$ (ranksum = 15). This implies that the SCOP benchmark has a better coverage of average substitution patterns for different levels of evolutionary divergence, thus resulting in a more generalisable model of sequence evolution. (Specifically, SCOP2 is nearly three times larger than SABMARK-Sup, and seven times that of HOMSTRAD). Between SCOP1 and SCOP2, the difference lies in the alignment hypotheses for the same set of protein domain pairs. The general trend observed for $\text{MML}_{\text{SCOP1}}$ continues for $\text{MML}_{\text{SCOP2}}$ as well. However, all total message length figures reported for the SCOP DALI alignments are significantly higher (despite the smaller number of alignments compared to SCOPMMLigner), indicating that the MMLigner structural alignments convey more protein *sequence* compressibility.

The sole outlier amongst the MML-inferred matrices was $\text{MML}_{\text{SABMARK-Twi}}$ with a ranksum = 42. As already stated, SABMARK-Twi benchmark contains alignments of highly-diverged sequence-pairs (i.e. an average sequence identity of ∼8.4%), providing an extremely weak sequence signal to infer a stochastic matrix which can be effectively generalised to explain a wider range of sequence relationships that any other benchmark may embody. However, a noteworthy observation is that, $\text{MML}_{\text{SABMARK-Twi}}$ (ranksum = 42) is nearly in par with PFASUM (ranksum = 41), the best performer amongst the set of existing matrices.

Shannon Information Content of Benchmarks under Choice 2

Choice 2 applies the stationary distribution of the stochastic matrix as the *null model* probabilities of amino acids for indels. Table 6.3 presents the total message lengths resultant for each existing matrix and inferred matrix across the benchmarks, with their respective `ranksums`. In contrast to choice 1 results, Table 6.3 reflects how well each stochastic matrix explains the *null model* distribution of amino acids in protein sequences through their stationary distributions. Under this category, each MML-inferred matrix is a resultant of an optimisation that included its stationary distribution to explain the indels in the respective benchmark. The order of compression ability changes, with the overall observation that, $\text{MML}_{\texttt{SCOP2}}$ matrix still performs the best across all benchmarks with the lowest `ranksum` = 16. This time, $\text{MML}_{\texttt{SABMARK-Sup}}$ has beaten $\text{MML}_{\texttt{SCOP2}}$, becoming the next best to $\text{MML}_{\texttt{SCOP1}}$. Note that overall message lengths are higher than when using choice 1 (this is again apparent since choice 1 optimised the null distribution for the particular benchmark).

Shannon Information Content of Benchmarks under Choice 3

Choice 3 simply applies optimal *null model* estimates inferred over protein sequences from the UniProt database (as discussed in §4.4). This 20-state distribution $\mathbf{P}$ is no longer communicated in the respective encoding message, as it is simply common knowledge. See Table 6.4 for results. Here, the `ranksums` observed for choice 1 in the Table 6.2 are preserved, except for `LG` and `MIQS` where their ranks over `SCOP2` have interchanged.

Altogether, the 20-state *null model* estimates inferred on indel regions of the benchmark gives a better (i.e. shorter) encoding length, compared to that of when using the stationary distribution of the substitution matrix. The $\mathbf{P}$ inferred on UniProt gives the longest encoding length amongst the three choices. Refer Figure 6.4 for overall trends in total encoding lengths of benchmarks across the fifteen matrices. Table 6.5 summarises the results in terms of matrix `ranksums` across the three choices for $\mathbf{P}$. A notable observation is that, `PAM`, `JO` and $\text{MML}_{\texttt{SCOP2}}$ `ranksums` remain the same across the three different choices for $\mathbf{P}$.

Conclusions

This section presented a unique comparative analysis of an important set of existing substitution models, which undoubtedly signifies the continuous improvement of amino acid substitution modelling over time. The MML matrix optimisation framework facilitates their consistent and fair evaluation. Broadly, the performance of non-Markov models converted into proper Markov models favours the hypothesis of a time homogeneous Markovian behaviour in protein sequence evolution (consistent with how substitution matrices were originally conceived). The framework supports any non-Markov, single substitution matrix to be utilised as a Markov matrix, which effectively accounts for the optimal evolutionary time between two proteins when their relationship is evaluated. The $\text{MML}_{\texttt{SCOP2}}$ stochastic matrix inferred over the `SCOP2` benchmark outperforms all the other matrices, displaying a `ranksum` = 15 and `ranksum` = 16 under both choices 1,3 and choice 2, respectively. Hence, it is chosen as a newly inferred, reasonable Markov model of amino acid substitution. As already highlighted in its very first mention at the end of §6.3.2, this matrix is named as MMLSUM.

The next two sections specifically look at the similarities and differences between all nine existing matrices and the new matrix MMLSUM. Additionally, Appendix B illustrates different amino acid properties captured by those matrices. An independent focus on the characteristics and properties of MMLSUM is made in §6.5.

Table 6.3: Shannon information content (in bits) to losslessly encode each benchmark with varying substitution matrices under **choice 2**. Their ranks of performance are reported inside brackets. Note: **P** is the stationary distribution of the respective substitution matrix.

Matrix	HOMSTRAD	Mattbench	SABMARK-Sup	SABMARK-Twi	SCOP-1	SCOP-2
Shannon information using existing substitution models						
PAM (1978)	11547416.21 (15)	9164067.18 (15)	23657379.78 (15)	11355479.09 (15)	85072048.36 (15)	82949452.90 (15)
JTT (1992)	11496340.10 (14)	9084284.48 (13)	23499066.75 (13)	11273707.07 (11)	84446694.25 (13)	82346379.77 (13)
BLOSUM (1992)	11459479.32 (11)	9060438.92 (10)	23466561.57 (11)	11275166.62 (13)	84354008.80 (11)	82232019.26 (11)
JO (1993)	11490030.65 (13)	9139874.62 (14)	23571493.08 (14)	11328875.88 (14)	84680102.91 (14)	82560968.37 (14)
WAG (2001)	11432008.93 (07)	9065038.48 (11)	23442803.68 (10)	11262564.43 (10)	84218617.83 (10)	82107643.24 (10)
VTML (2002)	11439908.46 (10)	9049441.74 (07)	23432043.81 (09)	11255766.35 (09)	84180147.95 (09)	82070311.94 (09)
LG (2008)	11485493.34 (12)	9065759.17 (12)	23491875.49 (12)	11274788.00 (12)	84399024.43 (12)	82286691.96 (12)
MIQS (2013)	11435028.94 (08)	9051333.90 (09)	23421817.75 (08)	11252372.16 (08)	84147813.60 (08)	82031386.28 (08)
PFASUM (2017)	11423630.07 (02)	9048568.06 (06)	23406776.08 (05)	11248125.05 (06)	84095385.87 (04)	81984520.86 (04)
Shannon information using MML inferred substitution models						
$\text{MML}_{\text{HOMSTRAD}}$	**11418789.53 (01)**	9050681.63 (08)	23408123.19 (06)	11250120.55 (07)	84113701.63 (06)	82002419.36 (06)
$\text{MML}_{\text{MATTBENCH}}$	11436960.04 (09)	**9040249.80 (01)**	23416280.31 (07)	11248122.03 (05)	84130366.58 (07)	82027458.07 (07)
$\text{MML}_{\text{SABMARK-Sup}}$	11423841.26 (03)	9044793.01 (05)	**23401281.01 (01)**	11244464.40 (02)	84092356.97 (03)	81982211.04 (03)
$\text{MML}_{\text{SABMARK-Twi}}$	11427701.22 (06)	9043437.94 (02)	23404974.85 (04)	**11243567.19 (01)**	84099460.19 (05)	81990981.18 (05)
$\text{MML}_{\text{SCOP1}}$	11424651.38 (05)	9043895.49 (04)	23403924.44 (03)	11244987.34 (03)	**84075928.68 (01)**	81970279.92 (02)
$\text{MML}_{\text{SCOP2}}$	11423906.32 (04)	9043878.06 (03)	23403244.25 (02)	11245052.04 (04)	84077082.89 (02)	**81968197.28 (01)**

Table 6.4: Shannon information content (in bits) to losslessly encode each benchmark with varying substitution matrices under **choice 3**. Their ranks of performance are reported inside brackets. Note: **P** is the UniProt based *null model* probabilities of amino acids

Matrix	HOMSTRAD	Mattbench	SABMARK-Sup	SABMARK-Twi	SCOP-1	SCOP-2
Shannon information using existing substitution models						
PAM (1978)	11547347.32 (15)	9155559.27 (15)	23628852.13 (15)	11336736.12 (15)	85026098.24 (15)	82893663.55 (15)
JTT (1992)	11496994.08 (14)	9084491.06 (13)	23505598.27 (13)	11278424.47 (13)	84454677.89 (13)	82354250.08 (13)
BLOSUM (1992)	11453343.72 (10)	9049472.26 (08)	23428674.77 (07)	11254553.75 (06)	84275402.19 (11)	82130897.40 (10)
JO (1993)	11492309.39 (13)	9130688.40 (14)	23556128.16 (14)	11316566.45 (14)	84668169.16 (14)	82541280.05 (14)
WAG (2001)	11434976.91 (05)	9065145.32 (12)	23454783.64 (11)	11269752.08 (12)	84242325.21 (09)	82131872.38 (11)
VTML (2002)	11439289.05 (07)	9048325.84 (07)	23432271.63 (08)	11257134.26 (08)	84176599.86 (07)	82061020.74 (07)
LG (2008)	11480054.48 (12)	9061463.01 (11)	23466479.90 (12)	11261899.34 (09)	84356348.28 (12)	82226007.09 (12)
MIQS (2013)	11438006.26 (06)	9052903.26 (10)	23440009.45 (10)	11262833.45 (10)	84177434.21 (08)	82063425.69 (08)
PFASUM (2017)	11428679.05 (02)	9052221.77 (09)	23433841.03 (09)	11263082.39 (11)	84141210.67 (04)	82038431.69 (06)
Shannon information using MML inferred substitution models						
$\text{MML}_{\text{HOMSTRAD}}$	**11421395.62 (01)**	9047739.99 (05)	23419917.83 (06)	11256695.07 (07)	84126993.63 (03)	82009294.02 (03)
$\text{MML}_{\text{MATTBENCH}}$	11442135.03 (09)	**9038304.82 (01)**	23409982.48 (03)	11243729.20 (03)	84151618.78 (05)	82022514.41 (04)
$\text{MML}_{\text{SABMARK-Sup}}$	11439926.80 (08)	9043738.29 (04)	**23400792.07 (01)**	11238762.79 (02)	84168584.28 (06)	82024870.38 (05)
$\text{MML}_{\text{SABMARK-Twi}}$	11458572.58 (11)	9048142.90 (06)	23410821.13 (05)	**11237871.03 (01)**	84256392.91 (10)	82098025.94 (09)
$\text{MML}_{\text{SCOP1}}$	11429086.52 (03)	9042105.23 (03)	23410002.48 (04)	11247805.78 (05)	**84097487.36 (01)**	81984099.12 (02)
$\text{MML}_{\text{SCOP2}}$	11429516.19 (04)	9041089.95 (02)	23404593.45 (02)	11244715.72 (04)	84100227.75 (02)	**81976372.74 (01)**

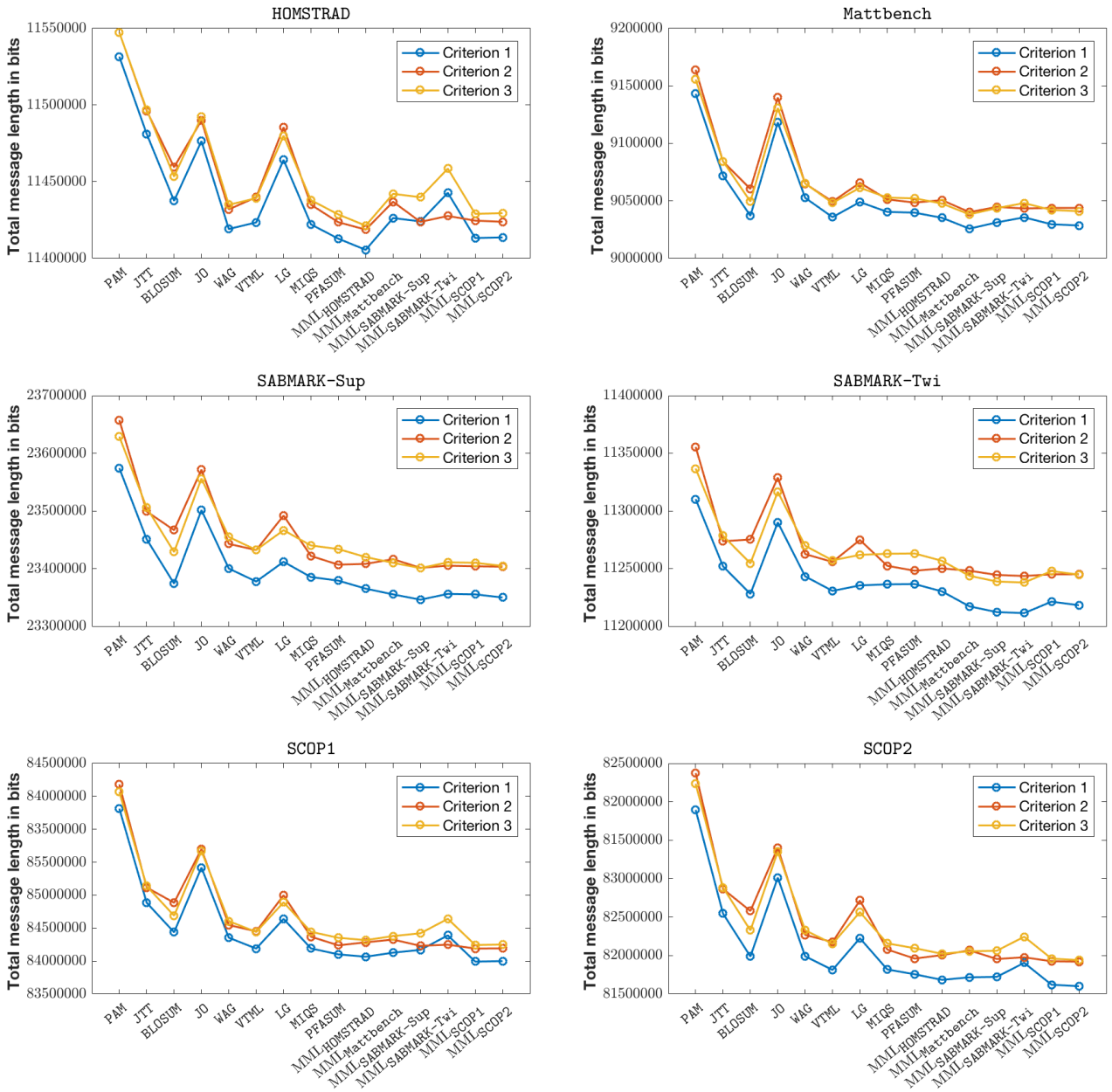


Figure 6.4: Encoding lengths (Shannon information content) of all six benchmarks resultant from different substitution matrices under the three different choices for **P** 20-state *null model* distribution

6.4.4 Kullback-Leibler Divergence between Matrices

This comparative study undertook an analysis to quantitatively highlight the similarities and differences between the ten stochastic matrices (nine existing matrices and MMLSUM (i.e. $\text{MML}_{\texttt{SCOP2}}$)) independent of any benchmarks. It should be emphasised that, this analysis *does not* provide any statement on the relative performance of various matrices (presented in the previous section), yet assists in gauging their relative concordance. Here, the notion of Kullback-Leibler (KL) divergence (Kullback and Leibler, 1951) is computed between any two substitution matrices. KL divergence estimates the measure of *relative* Shannon entropy between two probability distributions. For any two matrices X and Y (considered in their stochastic form), their

Table 6.5: Matrix `ranksum` under the three choices for $\mathbf{P}$ (along with their ranks inside parentheses)

Matrix	Choice 1	Choice 2	Choice 3
PAM	90 (14)	90 (15)	90 (13)
JTT	79 (12)	77 (13)	79 (11)
BLOSUM	52 (08)	67 (11)	52 (08)
JO	83 (13)	83 (14)	83 (12)
WAG	60 (10)	58 (10)	60 (09)
VTML	44 (07)	53 (09)	44 (07)
LG	64 (11)	72 (12)	68 (10)
MIQS	56 (09)	49 (08)	52 (08)
PFASUM	41 (05)	27 (05)	41 (05)
$\text{MML}_{\text{HOMSTRAD}}$	25 (03)	34 (06)	25 (03)
$\text{MML}_{\text{MATTBENCH}}$	25 (03)	36 (07)	25 (03)
$\text{MML}_{\text{SABMARK-Sup}}$	26 (04)	17 (02)	26 (04)
$\text{MML}_{\text{SABMARK-Twi}}$	42 (06)	23 (04)	42 (06)
$\text{MML}_{\text{SCOP1}}$	18 (02)	18 (03)	18 (02)
$\text{MML}_{\text{SCOP2}}$	**15 (01)**	**16 (01)**	**15 (01)**

KL divergence is measured in two different ways:

$$\text{KL divergence (over joint probabilities)} = \sum_{i=1}^{20}\sum_{j=1}^{20} X_{i,j} \log\left(\frac{X_{i,j}}{Y_{i,j}}\right)$$

$$\text{KL divergence (over conditional probabilities)} = \sum_{i=1}^{20}\sum_{j=1}^{20} X_{i,j} \log\left(\frac{X_{i|j}}{Y_{i|j}}\right)$$

Here, $X_{i,j}$ denotes the joint probability implied by matrix X for the pair of amino acids indexed by i and j (and similarly, for $Y_{i,j}$). On the other hand, $X_{i|j}$ denotes the conditional (substitution) probability implied by matrix X of an amino acid indexed by i, given an amino acid indexed by j (and similarly, for $Y_{i|j}$). Note, by default, the stochastic matrix is in its conditional form containing conditional probability terms between amino acids. The joint probability terms are computed as products of their conditional probability times the stationary probability.

Figure 6.5 presents heatmaps of the KL divergence measure between each pair of the ten matrices (in terms of both, conditional probability and joint probability) in comparison. (Note, KL divergence is not a metric, thus the resultant heatmap matrix is not symmetric). To highlight the relative-change in the concordance between matrices as they evolve with time, Figure 6.5 shows KL divergence computed between matrices at three time points. These time points are equivalent to all matrices such that, the estimated expected change in amino acids are all same. Accordingly, we can look at relative differences of these Markov models at $\{1\%, 50\%, 80\%\}$ expected change.

The numbers suggest that the distance between matrices widens and the differences between them become more apparent as their expected change increases. `PAM` and `JO` are radically different from other matrices, and this correlates with their poor performance observed in terms of their `ranksums` as seen in the previous section.

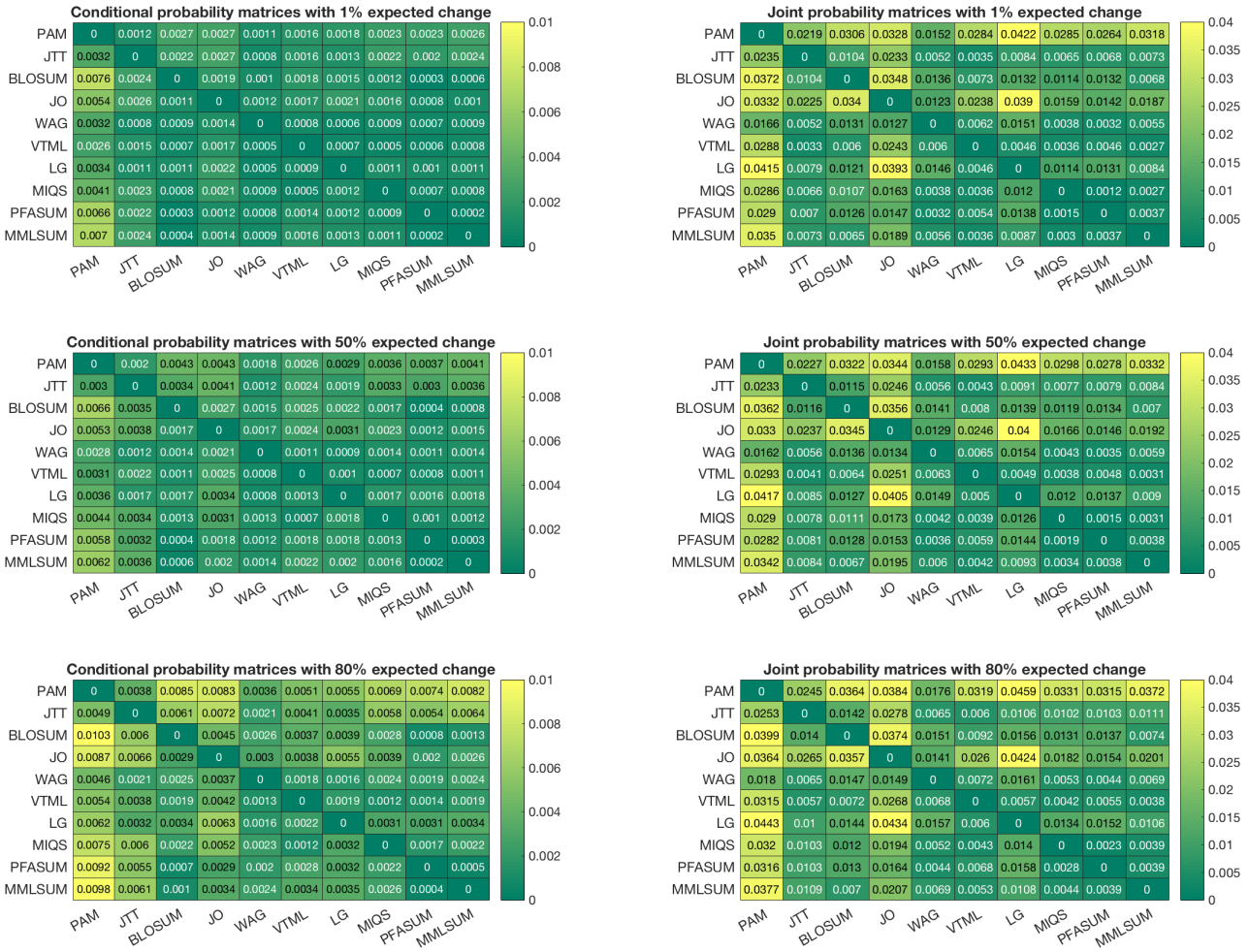


Figure 6.5: KL divergence based distance matrices (in the form of heat maps) between ten matrices (9 existing matrices and the newly inferred MMLSUM matrix). Left column is the computation of KL divergence over conditional probabilities of amino acid interchanges. Right column is the computation of KL divergence over the corresponding joint probabilities of interchanges. The three rows compare between equivalent states of matrices when their expected amino acid change is $1\%, 50\%$ and 80%, respectively. (Note: All measures are in nits)

Clustering of substitution matrices

Additionally, these heatmaps (i.e. distance matrices) of KL divergence can be used to cluster these substitution models and gain some insights on which of them are closest or farthest to each other. Figure 6.6 visualises dendrograms resultant from agglomerative hierarchical clustering of the ten conditional probability matrices and separately, their joint probability matrices, using the average linkage method (Unweighted average distance (UPGMA)) for computing the distance between clusters. Again, the expected change values of $1\%, 50\%, 80\%$ are considered.

A notable observation is that the clustering structures bear slight differences in their arrangement when the expected change increases. Also the clusters resultant from conditional matrices and joint matrices are clearly not in concordance. This is possible because, the joint probability model incorporates the matrix stationary distribution. Basically, matrix performance is reliant on the joint probability matrix, since a matched amino acid pair is encoded using its joint probability (i.e. conditional probability $\times$ stationary probability).

In clustering structures for joint probability matrices, `PAM` and `JO` are separated from the rest across all the three dendrograms. It reflects their similar weak performance in encoding an

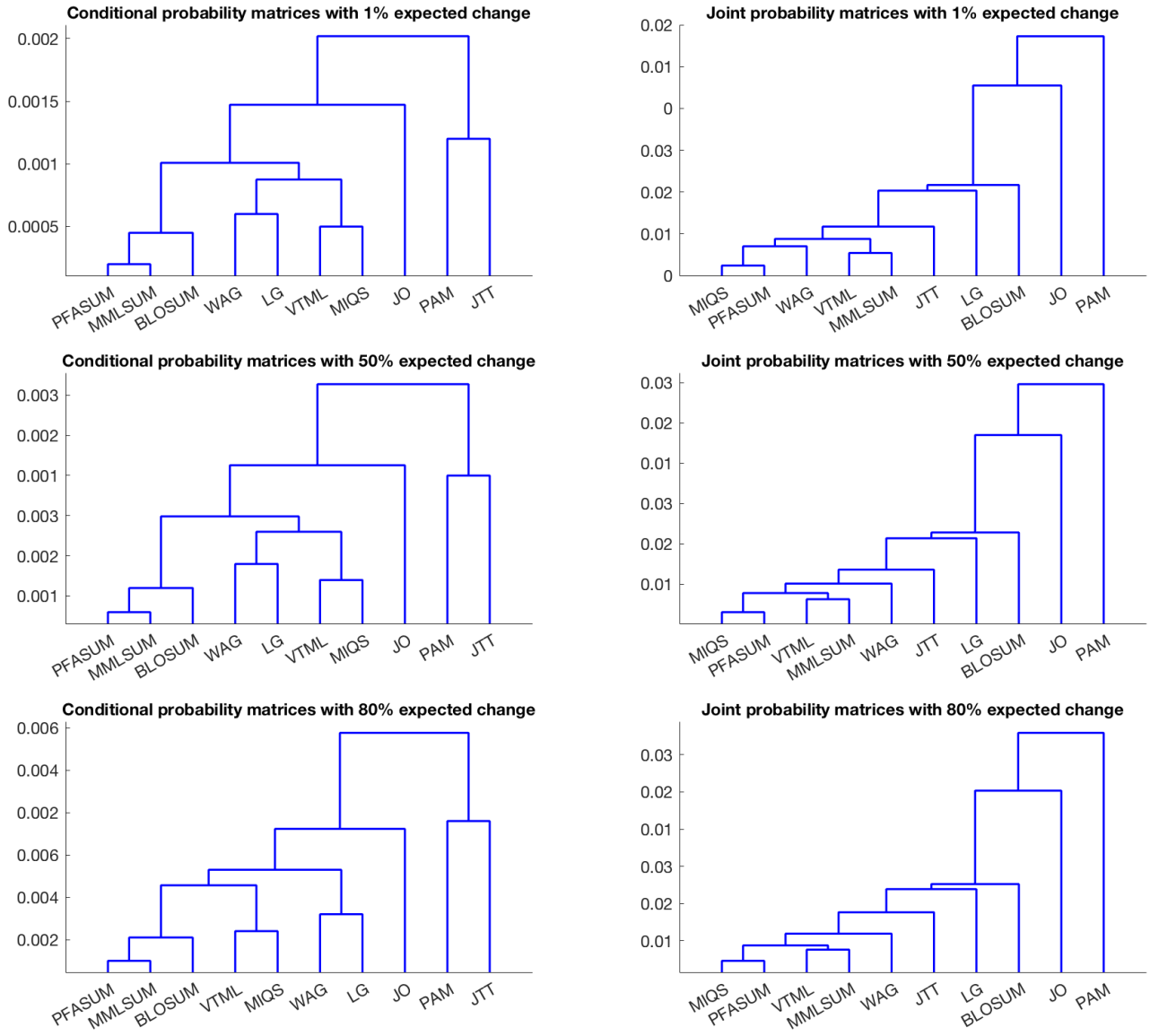


Figure 6.6: Agglomerative hierarchical clustering of conditional probability matrices (in the first column) and joint probability matrices (in the second column) based on their KL divergence matrix. Rows account for matrices with $1\%, 50\%$ and 80% expected change, respectively. Note: all KL divergence measures are in nits. (Note: clusterings were generated using MATLAB R2017a (The Mathworks, Inc., 2017))

alignment benchmark (as already observed in the previous section, `ranksum` $= 90$ for `PAM` and `ranksum` $= 83$ for JO, consistently across all three choices of $\mathbf{P}$ 20-state *null model* distribution). The next noticeable positions are of `LG` and `BLOSUM`. They are close performers with adjacent `ranksums`: 72 and 67, respectively under choice 2. Across all three expected change values, they are consistently separated out from the rest. Despite few differences in the clustering arrangement, `VTML`, `MIQS`, `PFASUM` and `MMLSUM` come under the same group, again reflecting their adjacent order of `ranksums` under choice 2. There, `MIQS` and `PFASUM` are consistently clustered the closest to each other, while `VTML` and `MMLSUM` makes another group.

Note: It is somewhat difficult to arrive at any direct correlations between matrix performance and matrix similarity, as it depends on the average similarity of all matrices ($\forall t \in [1, 1000]$) coming under each Markov model.

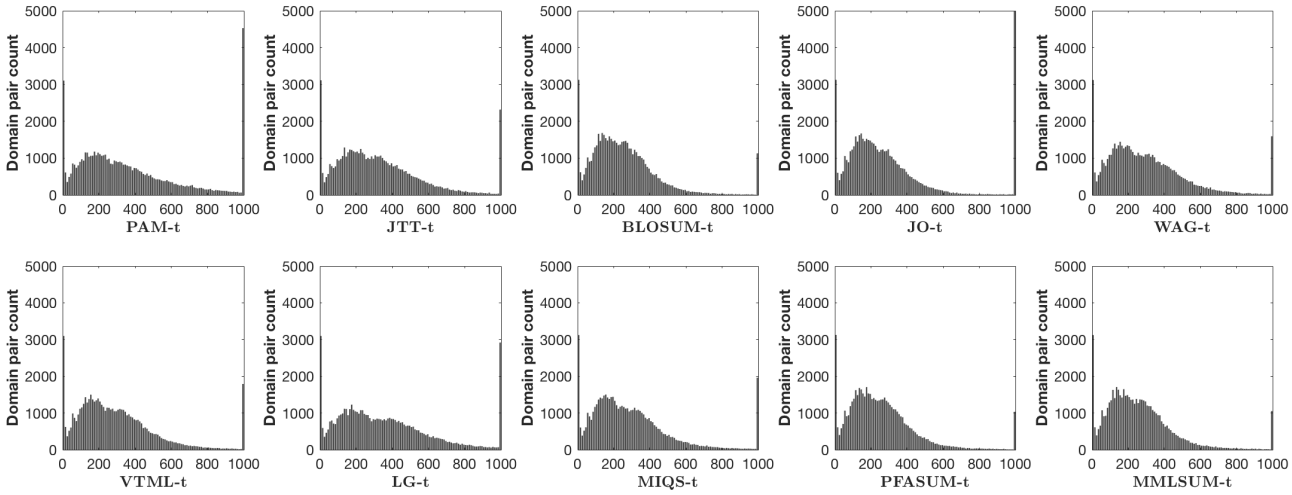


Figure 6.7: Histograms of the evolutionary time parameter t under each substitution model, inferred for the 59,092 aligned pairs of protein domains in the SCOP2 benchmark

In terms of the conditional probability matrices, PAM and its improved version JTT are grouped together, implying their proximity in the timeline of appearance. WAG and its improved version LG are closer under 1% expected change, yet seems to deviate away as the evolutionary time increases. On the other hand, VTML and MIQS which are close adjacent performers under choice 2 and 3, are also clustered together. As expected, PFASUM and MMLSUM are closer. Interestingly, BLOSUM is closest to both of them. Another interesting cluster is WAG and its improved version: LG. As expected, the latest PFASUM (published in 2017) matrix and the newly inferred MMLSUM matrix in this thesis (2020) are the closest to each other. Next to them is BLOSUM (published in 1990), reflecting the reason for the series to be still considered as timely.

Again, these groupings are interesting observations, yet do not necessarily inform relative performance characteristics of the matrices. Not all groupings are intuitively explainable.

6.4.5 Stochastic Matrix Convergence to Equilibrium

It is interesting to examine how the optimal evolutionary time t between two amino acid sequences may vary under different Markov models of amino acid substitution. Figure 6.7 illustrates the histograms of time t resultant under the ten different stochastic matrices, when they were used to encode the protein alignments in the SCOP2 benchmark. The variations present in the histograms are due to the different clock speeds these matrices possess (i.e. differences in the time taken to reach the equilibrium state of the Markov chain – the *mixing time*), despite having a base matrix (corresponding to $t = 1$ unit of time) that accounts for the same 1% expected change.

A decomposed view into the rate of convergence in terms of each amino acid can provide an insight to this (See Figure 6.8). KL divergence measure between each amino acid column in the conditional probability matrix and its stationary distribution reflects the speed in which each matrix column converges into its equilibrium state. Once an amino acid x column reaches equilibrium, it is no longer able to differentiate any substitution pair $x \to y$ against x and y random occurrences. This is the point where models find it difficult to detect any relationship between highly diverged sequences.

As observed in Figure 6.8, all columns of PAM takes a longer time for convergence compared to the others. Four outlier curves are present in its KL divergence plot, accounting for amino

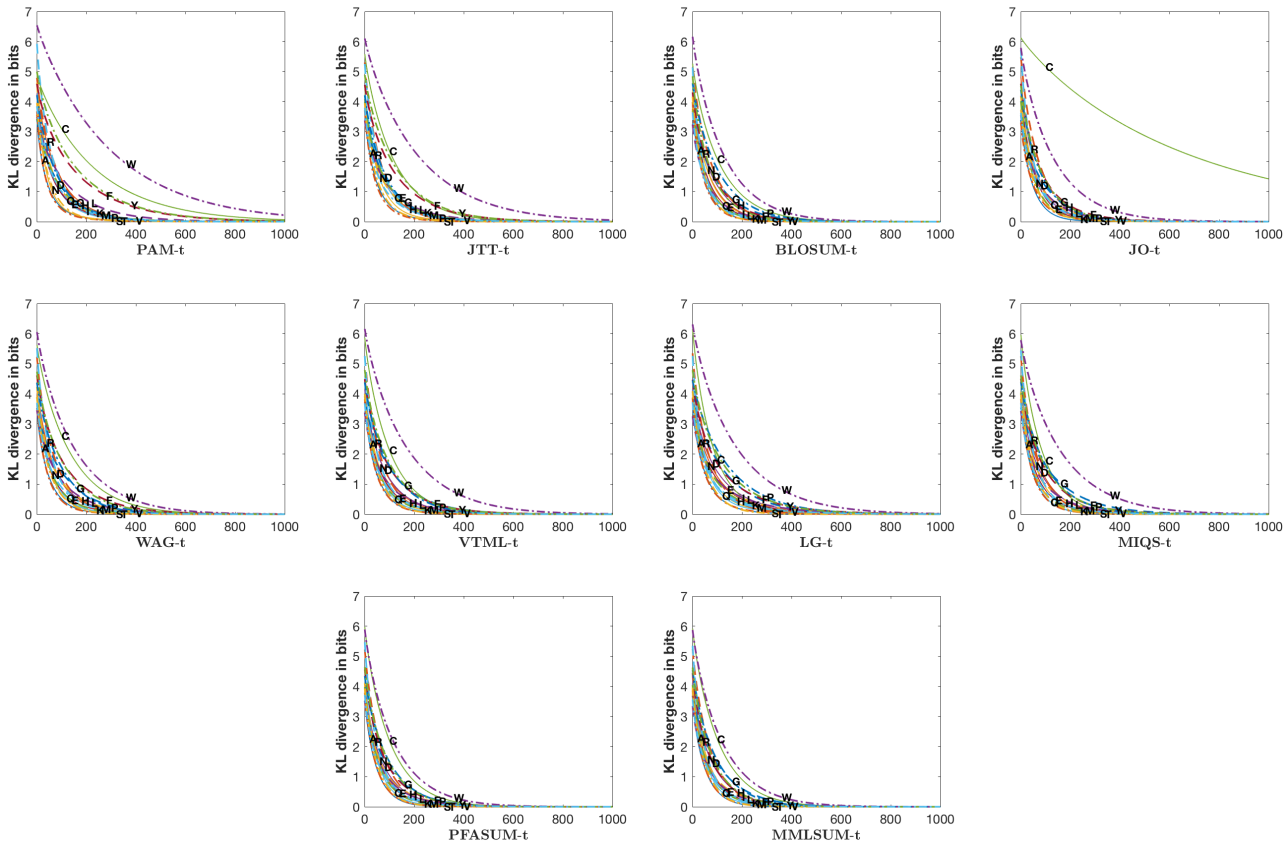


Figure 6.8: Across different substitution models, how each column vector of the conditional probability matrix reaches the matrix stationary distribution as evolutionary time t increases, measured in terms of the KL divergence between the two $\mathbb{L}1$-normalised probability vectors. Note how different matrices take a different time to reach the equilibrium state.

acids {W,C, F, Y}. Interestingly, only W acts as an outlier curve across all matrices, taking more time for convergence compared to other amino acids. C is not an outlier for LG and MIQS models. Note that C curve in JO is an extreme outlier with a significantly low convergence rate. The rest of the JO columns seems to converge more faster compared to PAM. This explains why we observe an optimal t distribution for JO in Figure 6.7 that is not as spread as PAM, JTT and LG. In general, all models have roughly reached equilibrium when $t = 1000$. Figure 6.9 zooms into the KL divergence plot of MMLSUM stochastic matrix. All curves roughly tend to 0 around $t = 600$, suggesting its limit of differentiation.

The difference in clock speed also relates to how the expected change of each model differs as a function of t, shown in Figure 6.10. The expected change of amino acid substitutions implicit in any substitution matrix shows to be non-linear with a logarithmic growth as the evolutionary time increases. This is due to the increase in the number of back substitutions that replace back the same amino acid after multiple time steps (i.e. $\mathtt{x} \to \mathtt{y} \to .. \mathtt{x}$). Eventually, the average observed percentage of difference asymptotically approaches a constant value as the matrix tend to the equilibrium state. At $t = 1000$, this value is 93.256% for PAM, while MMLSUM reaches 93.9497%. (See §6.2.3 on computing the expected change for a given conditional probability matrix of a Markov model).

The comparative study of amino acid substitution matrices presented in this section included representative matrices from the non-Markov series of BLOSUM and PFASUM, namely

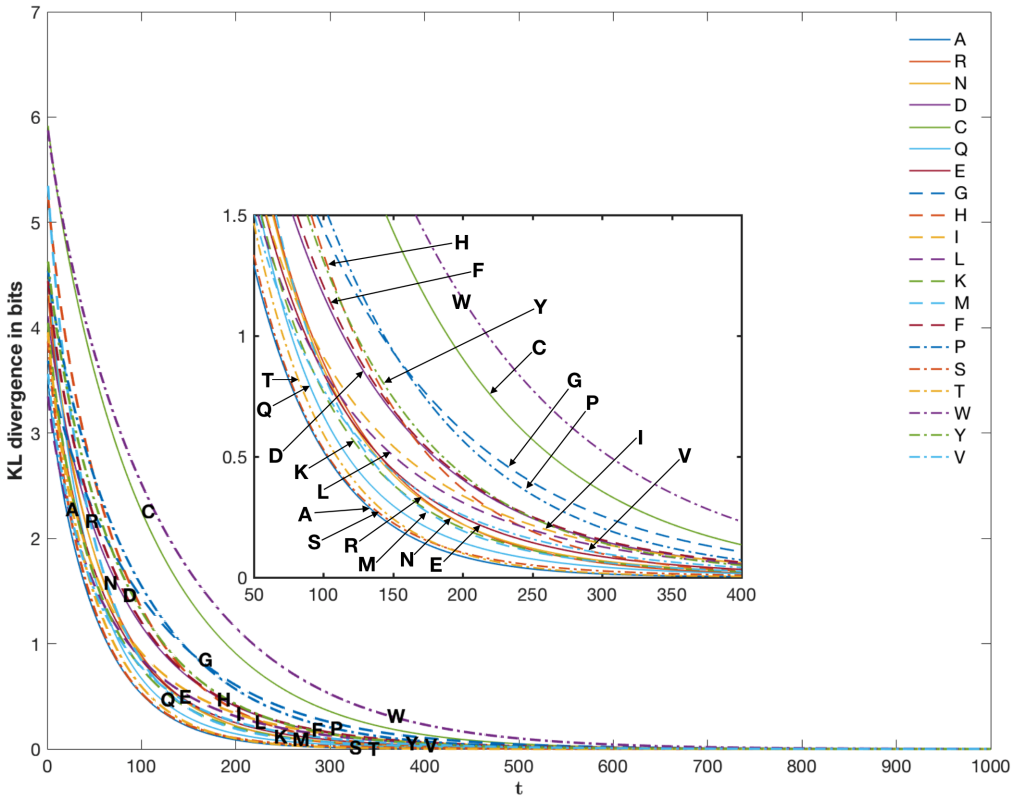


Figure 6.9: How each column vector of the MMLSUM conditional probability matrix reaches its stationary distribution as the evolutionary time t increases, measured in terms of the KL divergence between the two $\mathbb{L}1$-normalised probability vectors. Note that by $t = 600$, all columns are almost at equilibrium

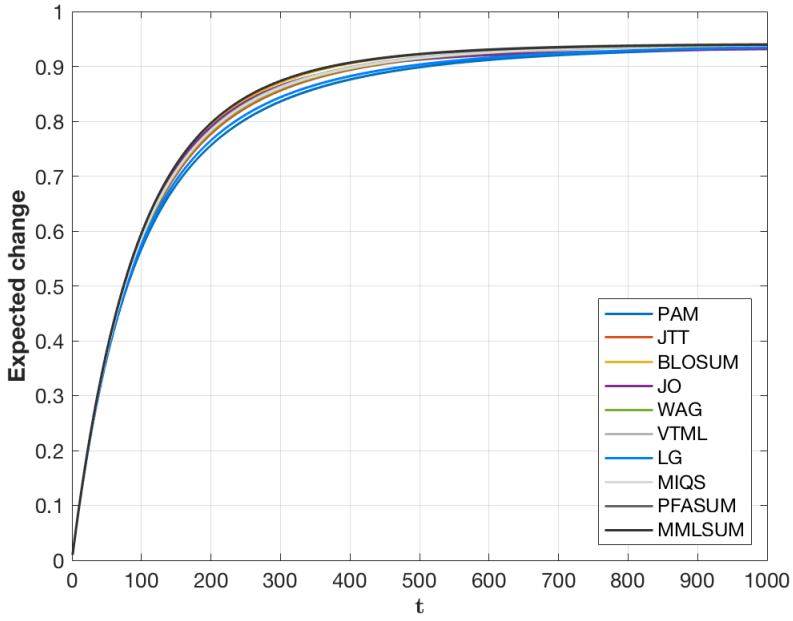


Figure 6.10: How the expected change of each Markov model changes as a function of evolutionary time t

BLOSUM-62 and PFASUM-60. The following auxiliary section extends the performance analysis across all equivalent matrices in these two series. The focus is to compare their Shannon information content of encoding the SCOP2 benchmark under **choice 1**.

6.4.6 Analysis of BLOSUM and PFASUM Series

This section accompanies an extended evaluation of the entire BLOSUM series (of sixteen matrices) and its equivalent PFASUM series, under choice 1 described in §6.4.3. Its purpose was to analyse how well all matrices in the BLOSUM series (i.e. most popularly used non-Markov series of substitution matrices) and PFASUM series (i.e. the best-performing, BLOSUM-like series amongst the existing substitution matrices) compress the SCOP2 benchmark, under their respective stochastic matrix form.

Table 6.6 is an extended set of results for Table 6.2 under the SCOP2 benchmark. Table 6.2 only conveyed Shannon information content in terms of a representative BLOSUM matrix (i.e. BLOSUM-62) and a representative PFASUM matrix (i.e. PFASUM-60). As reasoned previously, those two were chosen based on their applicability as a general-purpose log-odds scoring matrix in their respective series. Table 6.6 reports the Shannon information content associated with encoding all SCOP2 benchmark alignments using each BLOSUM-n matrix in the BLOSUM series, as well as each equivalent PFASUM-n matrix in the PFASUM series. The accompanying Figure 6.11 illustrates the total encoding message lengths (i.e. Shannon information content) reported in Table 6.6 as a function of the expected amino acid change implicit in their original scoring matrices. Consistent with the picture that emerged when comparing existing matrices, we can observe (see Figure 6.11) that the PFASUM series performs better than the BLOSUM series. Across the matrices in PFASUM, we can observe a flat trend of

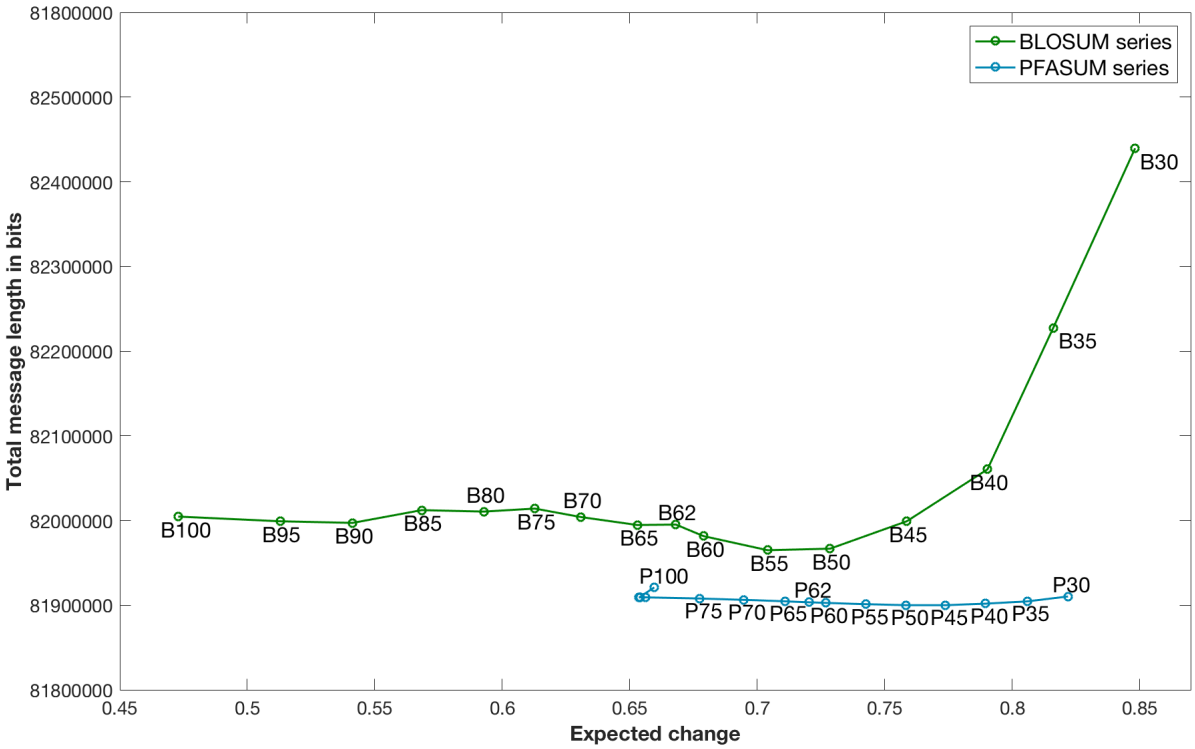


Figure 6.11: Shannon information content of each matrix in BLOSUM and PFASUM series versus their original expected change values (observed in their original non-Markov matrices). Bn and Pn are shortened names for BLOSUM-n and PFASUM-n, respectively.

Table 6.6: Shannon information content (in bits) to losslessly encode the SCOP2 benchmark using varying substitution matrices from the BLOSUM and PFASUM matrix series under **choice 1**

BLOSUM series	**Original expected change**	**Total message length**
BLOSUM-30	0.8482	82439452.76
BLOSUM-35	0.8161	82227426.03
BLOSUM-40	0.7903	82060377.07
BLOSUM-45	0.7587	81999170.77
BLOSUM-50	0.7286	81966740.66
BLOSUM-55	0.7041	**81964908.57**
BLOSUM-60	0.6790	81981508.57
BLOSUM-62	0.6681	81995179.31
BLOSUM-65	0.6530	81994670.34
BLOSUM-70	0.6309	82004034.57
BLOSUM-75	0.6128	82014056.31
BLOSUM-80	0.5928	82010530.24
BLOSUM-85	0.5684	82012220.79
BLOSUM-90	0.5413	81997158.82
BLOSUM-95	0.5129	81999074.38
BLOSUM-100	0.4729	82004612.48

PFASUM series	**Original expected change**	**Total message length**
PFASUM-30	0.8219	81910293.61
PFASUM-35	0.8059	81904481.06
PFASUM-40	0.7896	81901816.25
PFASUM-45	0.7739	**81899988.62**
PFASUM-50	0.7584	81900015.41
PFASUM-55	0.7427	81901163.14
PFASUM-60	0.7268	81902713.60
PFASUM-62	0.7204	81903374.10
PFASUM-65	0.7108	81904599.34
PFASUM-70	0.6946	81906335.59
PFASUM-75	0.6774	81907860.48
PFASUM-80	0.6562	81909272.54
PFASUM-85	0.6540	81909221.82
PFASUM-90	0.6538	81909091.84
PFASUM-95	0.6537	81909179.07
PFASUM-100	0.6596	81920990.78

encoding lengths suggesting their internal consistency, and that all of them are equally good to derive a stochastic matrix from. On the other hand, the BLOSUM series, with some variations, has a more or less flat trend when using matrices converted into a stochastic form within the range [45,100], whereas matrices in the range [30,40] give substantially worse (larger) message lengths in comparison. This suggests that BLOSUM[30-40] series of matrices do not generalise well to explain the range of relationships in SCOP2 benchmark. Another thing to note is that, BLOSUM-55 and PFASUM-45 are the best performers in their respective series.

BLOSUM being the most popularly used non-Markov family of substitution matrices, this section additionally carried out a short analysis to check the generalisability and representation of a BLOSUM-n based stochastic matrix with respect to all the other original BLOSUM matrices. First, Table 6.7 presents the KL divergence between original BLOSUM-n scoring matrices and their equivalent stochastic matrices, in terms of both conditional probability and joint probability. The k^{th} root of an original BLOSUM-n matrix $\mathbf{B}$ gives a base matrix $\mathbf{M}$. The t^{th} exponent of $\mathbf{M}$ is the matrix that has an expected change closer to that of $\mathbf{B}$. Ideally, we expect t to be exactly the same as k, and for the KL divergence to be 0. However, there are several sources of numerical approximation errors which obstruct that: (1) considering only integer time points, (2) normalisation of a matrix for columns to be adding up to 1, and (3) scaling a resultant base matrix to acquire 1% expected change. All matrices except BLOSUM-30, BLOSUM-35 and BLOSUM-40, have $k = t$ (as expected) with very low KL divergence values (reasonably approximating the ideal scenario), implying that the original matrix is well-represented by the derived Markov model. On the other hand, BLOSUM-30, BLOSUM-35 and BLOSUM-40 suffer from greater numerical instability. Having lower eigenvalues might also have contributed to this anomaly through matrix multiplication operations. This is likely the reason for BLOSUM[30-40] to behave as outliers with longer encoding lengths, compared to that of the other matrices in the series. Overall, BLOSUM-50 is the most well-represented as a Markov matrix. BLOSUM-62 (which is the series representative of BLOSUM in this thesis

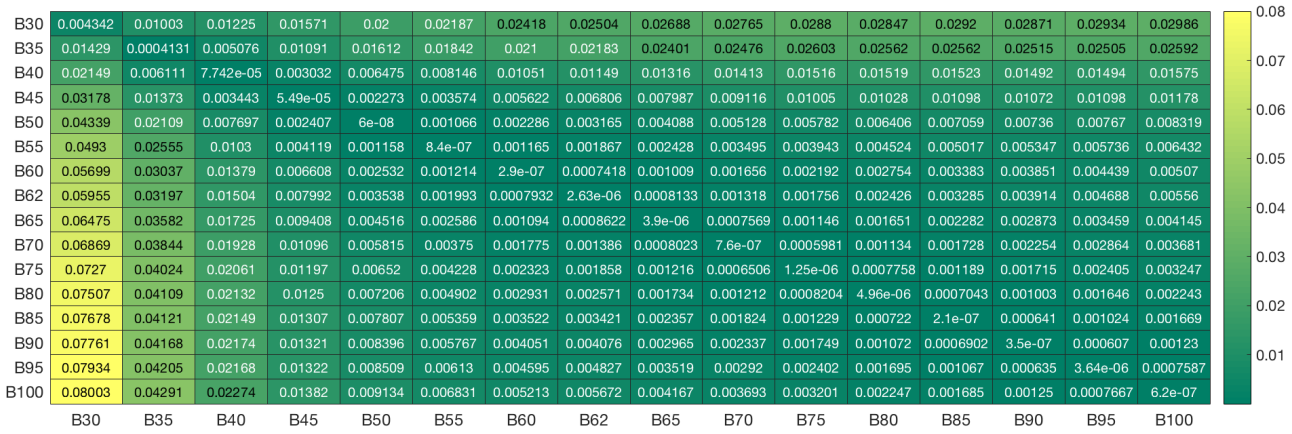


Figure 6.12: KL divergence between joint probability matrices of original BLOSUM-n (Bk) and their derived stochastic matrices corresponding to the original expected change. A row i corresponds to an original BLOSUM matrix i, whereas a column j corresponds to a stochastic matrix derived for an original BLOSUM matrix j and raised to some power t that gives an expected change approximately similar to that of BLOSUM matrix i. (Note: All KL divergence measures are in nits)

(due to its wide usage)) is also reasonably represented by its derived Markov model. A further comparison supports the conclusion that, under the integral precision, BLOSUM[30-40] matrices are not generalisable, while the rest in the series are reasonably closer in a Markov context. Figure 6.12 illustrates the heatmap of KL divergence between original BLOSUM matrices and their derived stochastic matrices equivalent to their original expected change. Note that the matrices in the same n (clustering percentage) neighbourhood tend to be more closer to each other.

While this analysis provides an insight into how BLOSUM matrices vary, a more conclusive study will be to derive the stochastic matrices with a higher precision and then evaluate them.

6.5 Characteristics and Properties of MMLSUM

This section explores characteristics and properties of the newly inferred Markov matrix of amino acid substitutions: MMLSUM. The MMLSUM matrix has shown to be effective in compressing any alignment benchmark, compared to the currently used substitution matrices (as evaluated in §6.4.3). Thus it can be taken as the most reliable representative of average amino acid substitution patterns observed across the present protein sequence repertoire. The conditional probability matrix (sometimes also known as the mutation probability matrix) of MMLSUM-1 ($t = 1$) base matrix is given by Table 6.8 (following a presentation similar to Dayhoff et al. (1978)). The diagonal elements clearly carry higher probabilities, as the expected change at this stage is just 1%.

6.5.1 Amino Acid Clustering

An amino acid substitution matrix encompasses average similarities and differences present within all possible amino acid substitution patterns, thereby capturing similar groups of amino acids. It can inform which amino acids are more frequently and less frequently changing, while giving more insights on their physico-chemical properties. This has been quite a known fact since the Dayhoff model of evolution. French and Robson (1983) appreciated the important

Table 6.7: How well an original BLOSUM-n scoring matrix (**B**) is reflected as a point (matrix **M**) in the derived BLOSUM-n based Markov model.

						KL divergence measure			
						Between $\mathbf{M}^t$ and **B**		Between $\mathbf{M}^n$ and **B**	
BLOSUM matrix (B)	**Expected change of B**	**k^{th} root of B**	**Expected change of $\mathbf{M}^k$**	**t^{th} exponent of M**	**Expected change of $\mathbf{M}^t$**	**Joint based**	**Conditional based**	**Joint based**	**Conditional based**
BLOSUM-30	0.8482	275	0.8660	249	0.8481	0.00434218	0.00434173	0.00654112	0.00654066
BLOSUM-35	0.8161	214	0.8198	211	0.8166	0.00041314	0.00041313	0.00050897	0.00050896
BLOSUM-40	0.7903	191	0.7912	190	0.7899	0.00007742	0.00007742	0.00008451	0.00008451
BLOSUM-45	0.7587	169	0.7586	169	0.7586	0.00005490	0.00005490	0.00005491	0.00005491
BLOSUM-50	0.7286	154	0.7285	154	0.7285	0.00000006	0.00000006	0.00000006	0.00000006
BLOSUM-55	0.7041	142	0.7036	142	0.7036	0.00000084	0.00000084	0.00000083	0.00000083
BLOSUM-60	0.6790	132	0.6794	132	0.6794	0.00000029	0.00000029	0.00000028	0.00000028
BLOSUM-62	0.6681	128	0.6691	128	0.6691	0.00000263	0.00000263	0.00000259	0.00000259
BLOSUM-65	0.6530	122	0.6538	122	0.6538	0.00000390	0.00000390	0.00000388	0.00000388
BLOSUM-70	0.6309	114	0.6315	114	0.6315	0.00000076	0.00000076	0.00000075	0.00000075
BLOSUM-75	0.6128	108	0.6135	108	0.6135	0.00000125	0.00000125	0.00000123	0.00000123
BLOSUM-80	0.5928	101	0.5914	101	0.5914	0.00000496	0.00000496	0.00000490	0.00000490
BLOSUM-85	0.5684	94	0.5687	94	0.5687	0.00000021	0.00000021	0.00000021	0.00000021
BLOSUM-90	0.5413	86	0.5409	86	0.5409	0.00000035	0.00000035	0.00000034	0.00000034
BLOSUM-95	0.5129	79	0.5142	79	0.5142	0.00000364	0.00000364	0.00000359	0.00000359
BLOSUM-100	0.4729	69	0.4723	69	0.4723	0.00000062	0.00000062	0.00000062	0.00000062

Table 6.8: Conditional probability matrix $\mathbf{M}$ of MMLSUM for evolutionary time $t = 1$ (describing a 1% expected change in amino acids). An element $\mathbf{M}_{ij}$ gives the probability that an amino acid a_j is replaced by an amino acid a_i after 1 unit of time. Elements are multiplied by 10,000, adhering to the same representation by Dayhoff et al. (1978). Columns denote the original amino acids, whereas rows denote replacement amino acids.

	A	**R**	**N**	**D**	**C**	**Q**	**E**	**G**	**H**	**I**	**L**	**K**	**M**	**F**	**P**	**S**	**T**	**W**	**Y**	**V**
A	9895	8	6	3	12	10	8	13	4	3	7	8	10	3	7	23	10	3	4	14
R	2	9895	6	2	1	13	7	0	9	1	3	25	3	0	1	5	5	2	3	1
N	1	3	9890	15	1	8	4	6	9	1	0	7	2	0	2	11	6	0	4	0
D	1	3	19	9913	0	8	20	5	5	0	0	6	0	0	4	7	4	0	1	0
C	1	0	0	0	9935	0	0	0	0	0	1	0	1	1	0	2	1	0	1	1
Q	5	9	6	3	0	9865	17	1	10	0	2	11	6	1	2	6	5	1	3	1
E	9	8	6	22	0	23	9895	1	5	1	1	15	2	0	5	7	7	0	2	2
G	10	4	8	6	3	3	3	9947	3	1	0	4	2	1	4	9	4	1	1	1
H	1	4	4	1	1	5	2	1	9908	0	1	3	1	1	1	2	1	1	7	0
I	4	2	1	0	4	1	0	0	1	9876	21	3	17	10	2	2	4	2	2	38
L	8	5	1	0	6	4	1	2	3	37	9909	2	39	21	3	2	4	8	5	19
K	6	29	11	7	0	18	14	1	5	1	2	9885	5	1	6	7	6	1	2	1
M	2	2	0	0	2	4	1	0	1	7	11	1	9864	3	0	1	2	2	3	2
F	2	1	1	0	3	0	0	1	3	6	11	0	8	9908	0	1	1	11	22	4
P	4	2	3	2	0	3	3	2	1	0	0	3	1	2	9936	4	3	0	1	3
S	16	6	16	11	6	9	7	9	6	1	0	7	5	2	9	9872	26	1	3	2
T	6	3	9	6	4	8	6	2	4	4	3	9	7	3	4	26	9885	2	3	9
W	0	1	0	0	0	0	0	0	1	0	1	0	1	3	0	0	0	9941	5	0
Y	2	4	3	1	2	2	1	0	10	1	2	1	4	19	1	2	3	10	9914	2
V	14	2	1	0	9	4	2	0	2	50	15	1	12	9	2	3	14	3	5	9890

information captured by PAM (Dayhoff et al., 1978), in terms of five observed amino acid groups: *aliphatic*, *aromatic*, *basic*, *hydrophilic* and *sulphydryl*. A multidimensional scaling over PAM has shown an amino acid group's tendency to be involved in a different secondary structural form (French and Robson, 1983). Jones et al. (1992b) noticed that the general trends of size and hydrophobicity conservation in both PAM and JTT matrices are similar, while their relative frequencies and mutabilities almost agree.

In this way, checking the clusters of amino acids implicit in a substitution matrix has been historically flagged as a viable method of checking if the particular matrix captures sensible patterns of amino acid substitutions. Hence, this section analyses the amino acid groups implicit in the newly inferred stochastic matrix, MMLSUM, using (1) hierarchical clustering and (2) tSNE embedding.

Figure 6.13(a) gives a dendrogram generated using agglomerative hierarchical clustering (based on average-linkage) over the MMLSUM base stochastic matrix (at $t = 1$). Following distinct groups can be identified:

- **Hydrophobic amino acids:**
  Valine (V), Isoleucine (I), Leucine (L), Methionine (M)
- **Aromatic amino acids:**
  Tryptophan (W), Tyrosine (Y), Phenylalanine (F)
- **Neutral amino acids:**
  Alanine (A), Serine (S), Threonine (T), Glycine (G), Proline (P)
- **Large amino acids:**
  Arginine (R), Lysine (K), Asparagine (N), Aspartic acid (D), Glutamic acid (E), Glutamine (Q)
- The remaining two amino acids, Histadine (H) and Cysteine (C) cluster apart from the rest.

To study the groupings observed more systematically and from a different point of view, we can apply the technique of t-distributed stochastic neighbor embedding (tSNE) (van der Maaten and Hinton, 2008) to MMLSUM. tSNE performs a non-linear dimensionality reduction of high-dimensional feature space and gives visualisations that aid detection of clustering in lower dimensions (Platzer, 2013; Li et al., 2017).

Currently, there exists different amino acid classification schemes that can be used as references to test how well a substitution matrix encapsulates various properties of amino acids. For the analysis here, this thesis takes distinct classes of amino acids listed mainly by Lefranc et al. (2015) (IMGT) under the following different classification schemes:

- Hydropathy (hydrophobic/hydrophilic/neutral)
- Charge (uncharged/positively charged/negatively charged)
- Polarity (non polar/polar)

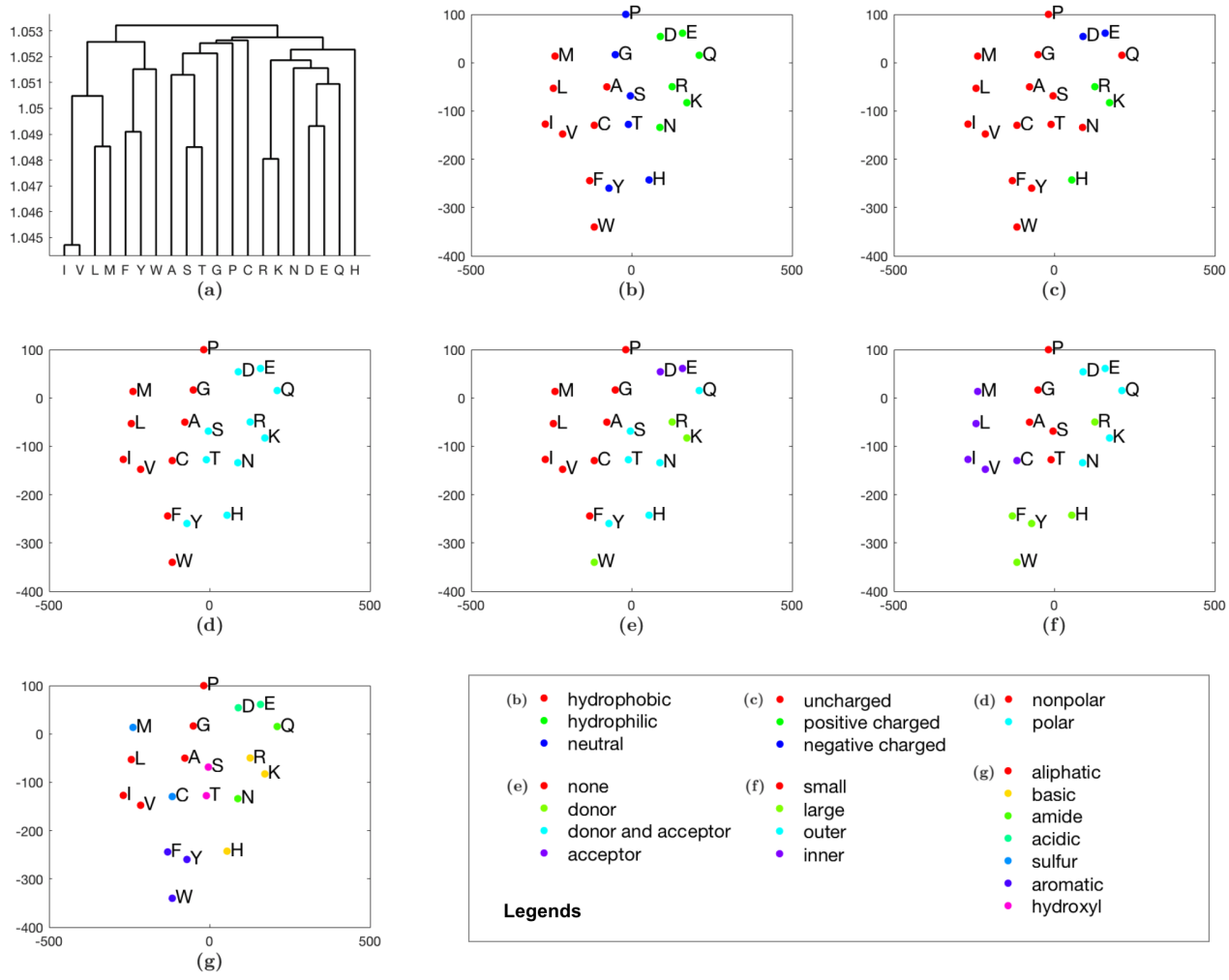


Figure 6.13: (a) Average-linkage clustering of amino acids generated from MMLSUM. (b)-(g) tSNE clustering of amino acids generated from MMLSUM. All plots have the same clustering, but coloured under different amino acid classification schemes based on: (b) hydropathy, (c) charge, (d) polarity, (e) hydrogen donor or acceptor, (f) volume & exposure and (g) chemical properties (based on IMGT classification (Lefranc et al., 2015) and (Swanson, 1984)). Refer to the legend for relevant colouring schemes in various subplots. (Note: clusterings and tSNE plots were generated using MATLAB R2017a (The Mathworks, Inc., 2017))

- Hydrogen donor or acceptor atom (none/donor/donor and acceptor/acceptor)
- Chemical (aliphatic/basic/amide/acidic/sulfur/aromatic/hydroxyl)

and the below scheme by Swanson (1984):

- Volume and exposure (small/large/outer/inner)

Figure 6.13(b)-(g) all show the same two-dimensional tSNE-visualisation of amino acids from MMLSUM, but each subplot colours the amino acids differently, based on the above listed widely-used amino acid classification schemes. These different schemes encompass the hydropathic character of amino acids, their charge, their polarity, their donor/acceptor roles in forming hydrogen bonds, their size and propensity for being buried/exposed, and their chemical constitution.

In Figure 6.13(b)-(f), the visualisation yields clearly separable amino acid groups on tSNE's 2D embedding of MMLSUM. In Figure 6.13(g) which deals with the classification based on the chemical characteristics of amino acids (as per IMGT (Lefranc et al., 2015)), the classes are mostly well-differentiated, barring a few outliers that include Histadine (H), Cysteine (C) and Asparagine (N) – Note that H and C are also outliers in the hierarchical clustering (cf. Figure 6.13(a)). See Appendix B for tSNE plots of the other existing substitution matrices. Amidst some variations, they all capture most of the amino acid groupings in general.

6.5.2 Amino Acid Divergence

Looking at the expected change trend displayed by the MMLSUM matrix (See Figure 6.10 or 6.14(a)): Previous studies (Rost, 1999) have shown that protein sequence relationships are most reliable when their sequence identity is $> 40\%$ (or the expected amino acid change is $< 60\%$). This corresponds approximately to the evolutionary time $t \in [1, 100]$ of MMLSUM. The 'twilight zone' of sequence relationships has been characterised by relationships sharing $[20 - 35]\%$ sequence identity (or $[65 - 80]\%$ change). This corresponds approximately to the range $t \in [125 - 200]$ of MMLSUM. Expected change of 90% is reached at $t = 400$, which then increases very slowly thereafter ($\sim 94\%$ change at $t = 1000$).

6.5.3 Insights on Matched Block Lengths and Gap Lengths

Unique to the MML protein alignment framework introduced in this thesis is the unified treatment of substitutions and gaps via time-parameterised Dirichlet models (See Chapter 5). §6.3.2 derived a set of 1-simplex and 2-simplex Dirichlet models for evolutionary time $t \in [1, 1000]$ under the newly inferred matrix, MMLSUM. (Recall Figure 6.2 which visualises those models as a function of evolutionary time). These Dirichlet models enable the analysis of expected matched block lengths and gap lengths in protein alignments, as a function of evolutionary time (Note: This is in the same way as we previously analysed for PAM Markov model in §5.3.5).

As previously explained in Chapter 5, the Dirichlet distributions model time-specific state transition probabilities of the alignment three-state machine over `match (m)`, `insert (i)`, and `delete (d)` states. Accordingly, there are nine transition probabilities involved in the alignment three-state machine (Figure 4.1), of which three are free (Pr(`m`|`m`), Pr(`i`|`i`) and Pr(`m`|`i`)), and the remaining dependent. In the three-state machine, the probability of moving from a `match` state to another `match` state (Pr(`m`|`m`)) controls the run length of any matched block in an alignment. The expected value of this run length, a geometrically distributed variable, is

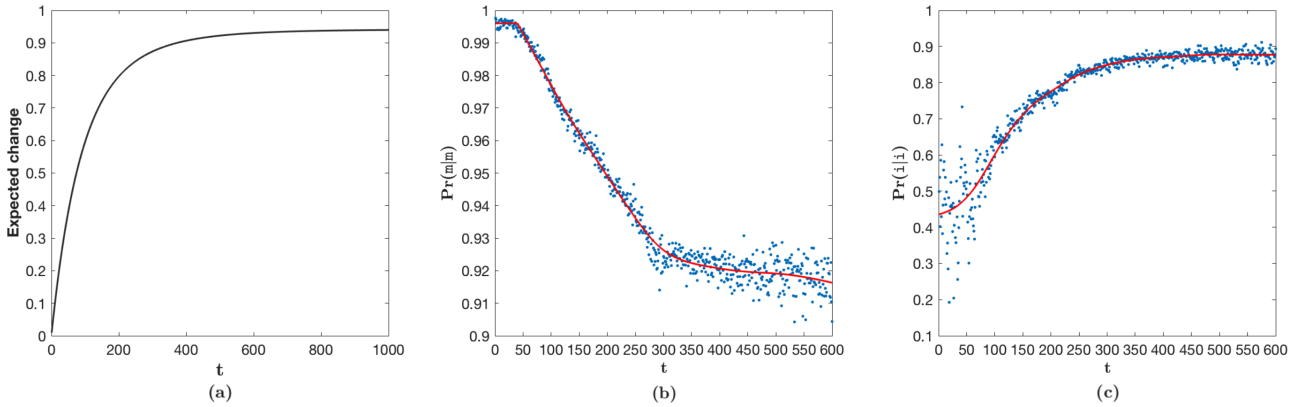


Figure 6.14: (a) Expected change of amino acids under the MMLSUM's model as a function of divergence-time parameter t (b) The variation of Pr(m|m) when derived from the *mean* of the inferred time-dependent Dirichlet distributions accompanying MMLSUM (Note: the mean can be taken as the expected value under each Dirichlet. They were smoothed as in the plot, using smoothing spline under the curve fitting tool in MATLAB R2017a (The Mathworks, Inc., 2017)) (c) Similarly, the variation of Pr(i|i) estimate with t. The divergence time parameter t is plotted in the range $[1, 600]$, beyond which the amino acids have near-converged to the stationary distribution of MMLSUM.

given by $\frac{1}{1-\Pr(\mathtt{m}|\mathtt{m})}$. Also, the value $1 - \Pr(\mathtt{m}|\mathtt{m})$ gives the probability of a gap (i.e. a block of insertions or deletions of any length) starting at a given position in an alignment.

Figure 6.14(b) plots the values of Pr(m|m) derived from the mean values of the inferred Dirichlets for the m state. We observe that it remains nearly a constant (Pr(m|m)= 0.9958 on average) in the range of $t \in [1, 40]$. This value corresponds to an *expected* run length of ~ 238 amino acids per block of matches. Sequence-pairs whose time parameter is in that range are closely-related, with $> 67\%$ amino acids expected to be *conserved* (cf. 6.14(a)) The probability of opening a gap ($1 - \Pr(\mathtt{m}|\mathtt{m}) = 0.0042$) for sequence-pairs in this range is extremely small. Next in the range $t \in [40, 300]$, Pr(m|m) decreases linearly with t. Comparing this range in the expected change of amino acids, we can see that it drastically increases from $\sim 32\%$ to $\sim 87\%$. This correlates with the expected length of match-blocks dropping from 238 amino acids to about 13. Further, for $t \geq 300$, Pr(m|m) decreases only gradually.

Similarly, the free parameter Pr(i|i) (equivalent to Pr(d|d) in the *symmetric* alignment state machine) controls the run lengths of indels. Figure 6.14(c) gives values of Pr(i|i) derived from the mean values of the inferred Dirichlets for the i state. In the range $t \in [1, 50]$, Pr(i|i) is noisy as the probability of observing a gap is small. Hence, there are only few observations of gaps from which to estimate this parameter. However in the range of $t \in [50, 400]$, Pr(i|i) grows from 0.5248 to about 0.8431, beyond which the probability flattens out at about 0.8759 on average (expected gap length $\frac{1}{1-\Pr(\mathtt{i}|\mathtt{i})} =\sim 8$ amino acids). The change of Pr(m|i) with t mirrors the behaviour of Pr(i|i). (Note, $\Pr(\mathtt{i}|\mathtt{i}) + \Pr(\mathtt{m}|\mathtt{i}) + \Pr(\mathtt{d}|\mathtt{i}) = 1$ and Pr(d|i) remains very small).

6.5.4 Functional Similarity Analysis

This section presents a complementary analysis on how functional similarity correlates with the evolutionary time between two proteins under the MMLSUM model of amino acid substitutions. Taking the main benchmark of 59,092 SCOP domain pairwise alignments (SCOP2), this study made use of the similarities between Gene Ontology (GO) (Ashburner et al., 2000; Gene

Ontology Consortium, 2004) tags of those protein domains as indicators of their functional similarity.

GO provides function annotations in terms of Biological Process, Molecular Function and Cellular Component. Biological Process (BP) is a higher level process, overarching multiple Molecular Functions (MF) that are carried out by proteins. Cellular Component (CC) refers to the location of a protein in action with respect to the cellular structure. For instance, the hemoglobin alpha subunit is annotated with the following GO tags: 'oxygen binding' and 'oxygen carrier activity' under MF; 'oxygen transport' and 'receptor-mediated endocytosis' under BP; and 'cytosol', 'extracellular exosome' under CC.

In this analysis, the GO tags of all protein domains in the SCOP2 benchmark were downloaded at first. However, there were some domains for which the tags were missing and thus, not all domain pairs could be included in the study. Only those where both domains have one or more of the above categories tagged in the GO database were considered. This resulted in 37,201 pairs for the exploration of functional similarity at the level of BP, with 48,215 pairs at the MF level, and 31,594 at the CC level.

The function similarity between two GO term lists (domain1 GO terms $\vec{x}$ versus domain2 GO terms $\vec{y}$) can be computed using their cosine similarity:

$$\text{Cosine Similarity } (\vec{x}, \vec{y}) = \frac{x.y}{||x||||y||}.$$

Figure 6.15 illustrates how the measure changes with evolutionary time.

The observations independently agree with MMLSUM's behaviour in reaching the equilibrium state with respect to the evolutionary time t (Recall Figure 6.9). As expected, the functional similarity measure decreases when the domains evolve further away from each other. More closely related proteins tend to have a similar set of functions; a trivial observation implied by the higher cosine similarity values that are closer to 1. However, values of t below 100 still

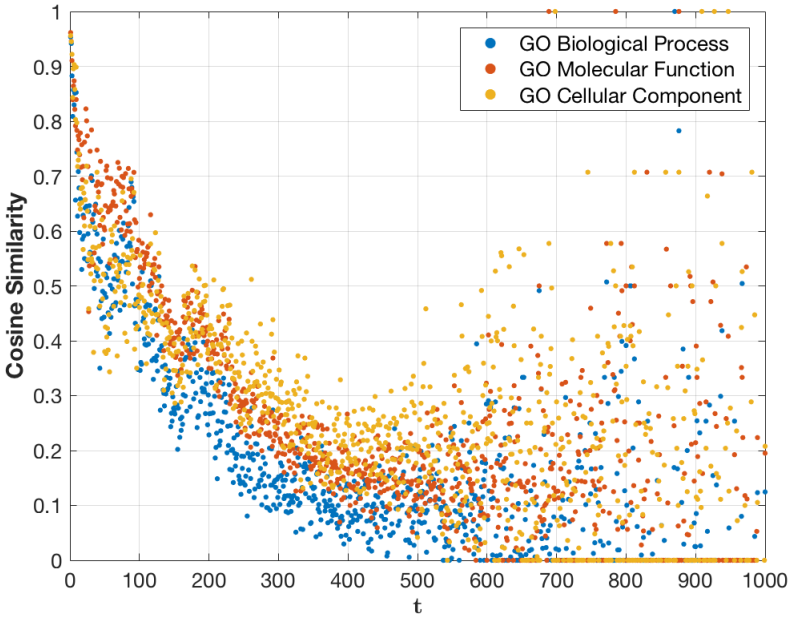


Figure 6.15: Cosine similarity between the Gene Ontology term vectors of protein domain pairs in the SCOP2 alignment benchmark, plotted against the evolutionary time parameter t of MMLSUM

seem to permit flexibility in acquiring different functions. Afterwards there is a clear curve of exponential decrease in functional similarity, reaching a somewhat noisy steady state. Beyond $t = 600$, the trend disappears, suggesting that, there can be many different functions acquired when domains have heavily diverged. This poses a practical upper limit on the applicability of the substitution matrix MMLSUM. (Note: Domain pairs got spread into only 962 discrete evolutionary time (MMLSUM-t) bins (38 time points are absent in the range $[773, 998]$).

Interestingly, studying the *'phylum'* (taxonomic rank) of each domain reveals the divergence of function from another point of view. Analysing the proportion of `SCOP2` domain-pairs that both belong to the same phylum by binning their inferred time parameters using MMLSUM, it was found that 92.3% of the domain-pairs whose inferred time parameters are in the range $t \in [1, 50]$ belong to the same phylum. Between $t \in (50, 100]$ this proportion falls to 53.5%. Roughly a similar proportion of 50.4% can be observed for values of $t \in (100, 200]$. Between $t \in (200, 300]$ and $t \in (300, 600]$, the number drops more drastically to 34.1% and 32%, respectively.

This concludes the analysis of the newly inferred MMLSUM Markov matrix of amino acid substitutions. The MML protein alignment framework introduced in Chapter 4 can now incorporate this substitution model in concert with its optimal evolutionary-time-parameterised Dirichlet models to align two protein sequences.

The following section concludes this chapter by presenting an example protein alignment generated under the MML protein sequence alignment framework with the new MMLSUM Markov matrix of amino acid substitutions alongside its evolutionary-time-parameterised three-state machines (based on the newly inferred Dirichlet models). Recall the *optimal alignment model* (§4.1) and *marginal probability model* (§4.2) discussed in Chapter 4. Out of curiosity, let us compare two interesting viral proteins that have been receiving attention in the year 2020 due to the ongoing global pandemic situation.

6.6 MML Sequence Alignment of an Interesting Pair of Proteins

Consider the alignment between *Spike glycoproteins* of the *Human SARS coronavirus* (SARS-CoV – PDB ID: `5WRG_1` (Gui et al., 2017)) and *SARS coronavirus 2* (SARS-CoV-2 – PDB ID: `7JZL_1` (Cao et al., 2020)) under the MML framework. The spike glycoprotein resides in the viral envelope of the coronavirus and initiates viral entry (and in turn infection) via host-cell attachment, binding to a host-cell receptor, and mediating membrane fusion of host-cell and virus (Gui et al., 2017). The below inferred *optimal alignment model* and *marginal probability model* clearly imply the close evolutionary relationship between the SARS-CoV (first appeared in 2003) and SARS-CoV-2 (novel coronavirus, first appeared in December 2019).[6]

The `5WRG_1` is an amino acid sequence of 1203 residues, while `7JZL_1` is of length 1288. Below lists the statistics resultant from running an *optimal alignment* and *marginal probability based alignment* under the MML framework.

[6]Source of dates: World Health Organisation https://www.who.int/ith/diseases/sars/en/ and https://www.who.int/csr/don/06-november-2020-mink-associated-sars-cov2-denmark/en/.

Table 6.9: Associated three-state machine parameters. (Note: Blue highlighted are the three free parameters of the state machine as illustrated by Figure 4.1. Others are dependent parameters.)

Parameter	**Estimate**$_{\text{Optimal}}$	**Estimate**$_{\text{Marginal}}$
Pr(m|m)	0.996	0.996
Pr(i|m)	0.002	0.002
Pr(d|m)	0.002	0.002
Pr(i|i)	0.453	0.450
Pr(m|i)	0.381	0.381
Pr(d|i)	0.166	0.169
Pr(d|d)	0.453	0.450
Pr(m|d)	0.381	0.381
Pr(i|d)	0.166	0.169

The *null model* message length: $I_{\text{NULL}}(\langle \mathbf{S}, \mathbf{T} \rangle)$	10545.862 bits
The *optimal alignment model* message length $I(\mathcal{A}^*, \langle \mathbf{S}, \mathbf{T} \rangle)$:	7586.996 bits
Compression ($\Delta I_{\text{Optimal}}$)	2958.870 bits
The *marginal probability model* message length $I_{\text{Marginal}}(\langle \mathbf{S}, \mathbf{T} \rangle)$:	7535.418 bits
Compression ($\Delta I_{\text{Marginal}}$)	3010.440 bits

- Inferred optimal evolutionary time (MMLSUM-t) = 27
- Inferred average evolutionary time (MMLSUM-t) = 24

Both models give positive compression. The inferred optimal evolutionary time suggests that the expected change of amino acids is about $\sim 23.25\%$ (i.e. these proteins are not very distant). On the other hand, the average $t = 24$ refers to an expected change of about $\sim 21.02\%$. Table 6.9 lists the associated three-state machine parameters for $t = 27$ and $t = 24$. Figure 6.16 visualises the marginal probability based alignment landscape of the two SARS viral proteins.

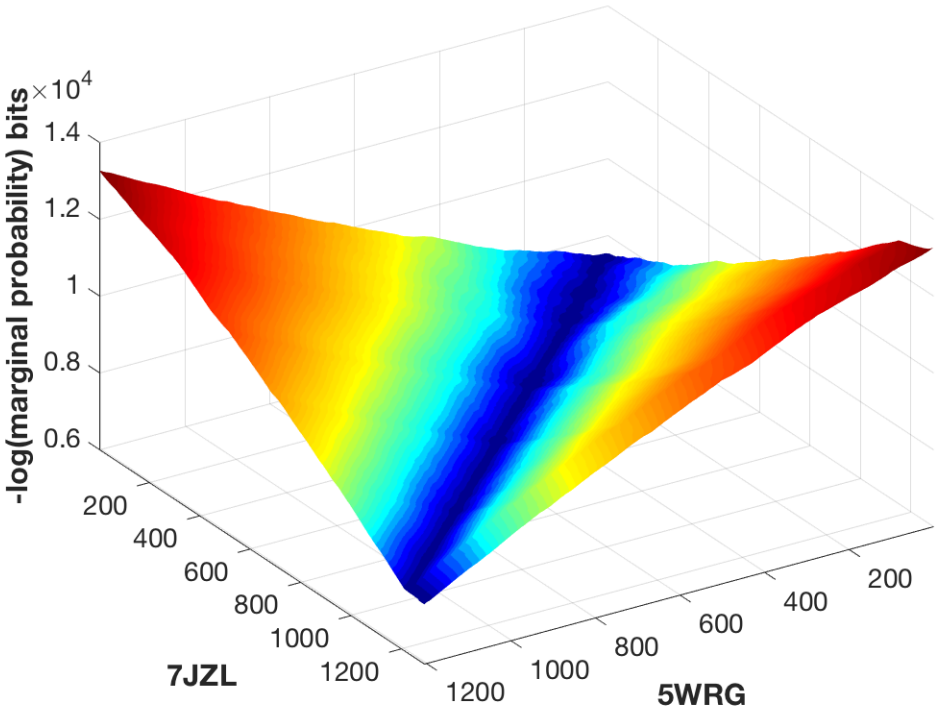


Figure 6.16: Marginal probability based alignment landscape for the Spike glycoproteins of the SARS-CoV and SARS-CoV-2 viruses

Chapter 7

Unified Protein Sequence-Structure Alignments

"All models are wrong, but some are useful"

– George E. P. Box

Previous chapters have developed the MML framework for protein *sequence* alignment along with a complete set of statistical models accompanying it. In this chapter, a proof-of-concept approach is presented to combine both sequence and structure information of proteins when searching for the best alignment relationship between them. The objective here is to infer a *sequence-structure alignment* by building on the outcomes presented in the thesis of Collier (2016), which provided an MML framework and separate models to address the *structure* alignment problem (i.e. solely using 3D coordinate information of atoms in proteins). To unify sequence and structure sources of information, this chapter uses a Naïve Bayes approach for combining sequence-based models (developed in this thesis) and structure-based ones (developed by Collier (2016)) to generate *sequence-structure* alignments of proteins.

7.1 Background

Combining sequence and structure sources of evidence to generate alignments is a computationally challenging task that requires, amongst other considerations, consistency within the models that account for both pieces of information. There have been previous attempts to utilise protein *sequence* and *structure* information jointly in protein alignment, for example in the work of Sali and Blundell (1990); Jones et al. (1992b); Levitt and Gerstein (1998); Buchan and Jones (2017). However, the field lacks a clear consensus on how to score and evaluate alignments that combine these two sources (Smith et al., 1997). Further, Yona and Levitt (2000) emphasise that:

> *"To achieve maximum sensitivity and benefit from both structures and sequences, we need to combine these two metrics. However, since these measures are based on different considerations it is not clear how one should judge scores that are assigned by either metric... Without knowing the relation between the two metrics it is hard to develop a notion of 'close' and 'far' which is consistent with both metrics. Moreover,*

it is not clear how statistical measures that are based on different protein features can be combined and transformed to a single scale."

Previous approaches have relied on assessing an independently generated sequence alignment and a structure alignment together, to arrive at a conclusion regarding the evolutionary relationship between the two proteins of comparison. A notable unification approach was presented by Levitt and Gerstein (1998) for assessing statistical significance of alignments based on the P-value/E-value statistic. As explained in §2.4.2, P-value statistic informs the probability of observing an alignment score of at least a certain value, under a *null* distribution of optimal alignment scores in the context of statistical significance testing. Such a null distribution can be estimated over unrelated protein alignments or artificially generated random protein sequence alignments.

For local, ungapped sequence alignments, the distribution of optimal alignment scores was reported by Karlin and Altschul (1990) to be a continuous probability distribution known as Gumbel, coming from the family of extreme-value distributions. Levitt and Gerstein (1998) separately confirmed that optimal structure alignment scores also follow an extreme-value distribution. Later, Bastien and Maréchal (2008) observed similar behaviour for global pairwise alignment scores as well.

The commonly used statistic (mostly in sequence database search of homologous proteins for a query protein) is the *E-value*. It is the expected number of random alignments that display a score greater than the particular score value under test. As mentioned previously in §2.4.2, the count of alignments with at least some threshold alignment score S is Poisson distributed. Thus, E-value is the mean under the Poisson distribution, computed as $k \cdot m \cdot n \cdot \exp(-\lambda S)$, where m and n are the query sequence length and subject sequence length, respectively, and k, λ are parameters to be estimated (Karlin and Altschul, 1990). The corresponding P-value is simply $1 - \exp(-\text{E-value})$. (Note: When it comes to database search (see§ 2.4.4), the length term n can simply be assigned the length of the entire database as a single long sequence).[1]

The E-value computation is different across different programs. For instance, BLAST (Altschul et al., 1990) relies on pre-computed values for k, λ specific for a given substitution scoring matrix and gap penalties, whereas FASTA (Pearson and Lipman, 1988) fits an extreme-value distribution for each sequence query, by comparing it against each sequence in the database. This process does not depend on an artificial null distribution, yet involves an effort to exclude protein pairs that are related. Nevertheless, it is a better strategy for dealing with composition bias of a query sequence (if present any) (Levitt and Gerstein, 1998).

As suggested above, the unification approach used by Levitt and Gerstein (1998) is noteworthy. Their experiment took a collection of SCOP domain pairs related at *superfamily* level (as true positives), and ran (1) standard sequence alignment (using Smith-Waterman local alignment under fixed gap penalties and BLOSUM50 substitution matrix), and (2) structure alignment (using a DP based approach), separately. Then they fitted a probability density function for sequence alignment scores over all-versus-all pairs that are true negatives. The same exercise was performed for structure alignment scores as well. Accordingly, the *E-value* statistics were computed for both types of alignment score. Levitt and Gerstein (1998) have compared sequence alignment significance against structure alignment significance using the resultant E-values, and identified a statistical significance threshold of 1% which agrees for both types of alignments. With this work, they substantiated *E-value* as a common ground to evaluate alignment scores across different programs (Levitt and Gerstein, 1998). This approach

[1]Details on these statistics are given by The National Center for Biotechnology Information at https://www.ncbi.nlm.nih.gov/BLAST/tutorial/Altschul-1.html.

has been later extended for classifying protein in terms of both their sequence and structure (Yona and Levitt, 2000).

Other approaches have also tried to incorporate sequence and structure. Recently, Fallaize et al. (2020) proposed a Bayesian approach that utilises a prior distribution for gaps in an alignment based on the sequence order to combine with the coordinate-based structure alignment. A related, yet a separate problem is *protein threading*, which aims to align an amino acid sequence to some known protein three-dimensional structure (Jones et al., 1992b). This is important for *homology modelling* where a computational search is performed to find the template structure (present in a library of known structures) that closely represents the true structure of a given sequence with unknown structure. Various scoring functions are being used in protein threading to accommodate both sequence and structure information such as amino acid residue-residue contacts and residue-site specific structural/energy characteristics (Bryant and Altschul, 1995; Buchan and Jones, 2017). A consistent unification of various *sequence* and *structure* level details can greatly assist protein threading. Lathrop et al. (1998) presented a Bayesian method for deducing the most probable alignment of a given sequence to a given core structure, with the aim of selecting the best core structure in a library. This is an instance of unifying core structure recognition and sequence-structure alignment.

Another effort is the recent version of MAFFT multiple sequence alignment tool called MAFFT-DASH (Rozewicki et al., 2019) which integrates sequence and structure using a pairwise structural alignment database called DASH. Two metrics: a residue-level structural similarity (based on the Gaussian function of the distance between C_α atoms of amino acid residues), and a domain-level similarity using a linear combination of sequence and structure based terms, are being used for this purpose.

7.2 MML based Sequence-Structure Alignment

Here we combine the MML framework and statistical models for protein *sequence* alignment (presented in this thesis) with the MML framework and models for protein *structure* alignment presented in the thesis of Collier (2016). Since these two works quantify the relationship between protein sequences and structures respectively in terms of strictly-additive Shannon information content, they can be combined in a straightforward fashion by adding the encoding lengths of sequence and structure data together. The below describes this in detail.

7.2.1 MML based Protein Structure Alignment

The thesis of Collier (2016) introduced the MML framework for protein *structure* alignment. Although the general framework has analogous information-theoretic considerations discussed in Chapter 4 (i.e. a well-defined *structure null model* and *structure alignment model*), the details between sequence models and structure models including the methods to search for the best alignment are necessarily and completely different.

Given a pair of proteins $\langle \mathbf{S}, \mathbf{T} \rangle$, let the ordered sets of C_α atomic coordinates for each residue of each linear amino acid chain be denoted by $\mathbf{S}_{\text{struct}} = \{\vec{s}_1, \vec{s}_2, \ldots, \vec{s}_{|\mathbf{S}|}\}$ and $\mathbf{T}_{\text{struct}} = \{\vec{t}_1, \vec{t}_2, \ldots, \vec{t}_{|\mathbf{T}|}\}$.

Structure null model refers to an independent encoding of these $\mathbf{S}$ and $\mathbf{T}$ structural coordinate data, with a total message length:

$$I_{\text{NULL}}(\langle \mathbf{S}_{\text{struct}}, \mathbf{T}_{\text{struct}} \rangle) = I_{\text{NULL}}(\mathbf{S}_{\text{struct}}) + I_{\text{NULL}}(\mathbf{T}_{\text{struct}}) \quad \text{bits} \tag{7.1}$$

Null encoding accounts for communicating the coordinates of any structure. In the original formulation of the null model for structures, Collier et al. (2014) used a uniform-direction encoding, which was later superseded by a mixture model of Kent distributions proposed by Kasarapu (2015). The encoding under the null model first states the count of C_α atoms in the structure over an integer code. Then the coordinates are encoded as follows. The (x, y, z) coordinate of each C_α is represented in a spherical coordinate system. Along the backbone of a linear amino acid chain, two consecutive C_α atoms are known to be distant on an average of 3.8Å. Using this information, the *null model* encodes any C_α coordinate given the previous three C_α coordinates, by transmitting the associated radius over a normal distribution with inferred mean and standard deviation, followed by its direction (relative to the previous three coordinates in a canonical orientation) using the mixture model of Kasarapu (2015) defined on the surface of a unit sphere.

Structure alignment model explains the atomic coordinate data of the two proteins according to some structure alignment hypothesis $\mathcal{A}_{\text{struct}}$ that describes how C_α atoms are related between the two proteins. The encoding lengths account for the alignment hypothesis $\mathcal{A}_{\text{struct}}$ and the C_α (x, y, z) coordinates of $\mathbf{S}_{\text{struct}}$ and $\mathbf{T}_{\text{struct}}$, yielding the total message length of the form:

$$I(\mathcal{A}_{\text{struct}}, \langle \mathbf{S}_{\text{struct}}, \mathbf{T}_{\text{struct}} \rangle) = I(\mathcal{A}_{\text{struct}}) + I(\langle \mathbf{S}_{\text{struct}}, \mathbf{T}_{\text{struct}} \rangle \,|\mathcal{A}_{\text{struct}}) \tag{7.2}$$

$$= \underbrace{I(\mathcal{A}_{\text{struct}})}_{\text{First part}} + \underbrace{I_{\text{NULL}}(\mathbf{S}_{\text{struct}}) + I(\mathbf{T}_{\text{struct}}|\mathbf{S}_{\text{struct}}, \mathcal{A}_{\text{struct}})}_{\text{Second part}} \quad \text{bits} \tag{7.3}$$

The first part accounts for encoding an alignment hypothesis as a three-state string over the `match` (`m`), `insert` (`i`) and `delete` (`d`) states using an adaptive Markov encoding scheme (and does not infer these using MML, as done in this thesis). The second part first sends $\mathbf{S}_{\text{struct}}$ using the *null model*, followed by an encoding for C_α coordinates of $\mathbf{T}_{\text{struct}}$ given the C_α coordinates of $\mathbf{S}$ and the alignment $\mathcal{A}_{\text{struct}}$, using probability distributions that account for local and global spatial similarities between the matched coordinates. The coordinates under insertion and deletion are stated using the null model (Collier, 2016).

Finally, any structure alignment hypothesis $\mathcal{A}_{\text{struct}}$ can be evaluated under the *structure alignment model* with respect to the *structure null model*, using the log odds posterior ratio that results in a compression statistic under the MML criterion as:

$$\Delta I_{\text{struct}} = I_{\text{NULL}}(\langle \mathbf{S}_{\text{struct}}, \mathbf{T}_{\text{struct}} \rangle) - I(\mathcal{A}_{\text{struct}}, \langle \mathbf{S}_{\text{struct}}, \mathbf{T}_{\text{struct}} \rangle) \quad \text{bits}$$

If $\Delta I_{\text{struct}} > 0$, we accept the structure alignment model, otherwise it is rejected.

Search method for the optimal alignment: Upon defining the above *alignment model* and *null model* for a given pair of proteins, the approach of Collier (2016) aims to find the optimal structure alignment using a two-phase heuristic search approach.

***Phase 1*:** *Seed alignment generation*

This involves an approach (Konagurthu et al., 2015) to produce a set of Maximal Fragment Pairs (MFP). An MFP is a maximally extended pair of substrings amongst the two protein structures that locally superposable within a threshold of RMSD value. These MFPs are filtered such that, each MFP is jointly superposable with at least two other MFPs in the filtered set. Next, MFPs are clustered into groups on which a heuristic seed alignment is generated for each cluster.

***Phase 2*:** *Local search alignment model optimisation*
This is a refinement step where, for each seed alignment, an EM based search is performed to minimise the total message length of the *alignment model* (i.e. *I-value*) by perturbing the alignment at each iteration.

7.2.2 Naïve-Bayes Unification

This section discusses the integration of MML based amino acid sequence models with the MML based 3D structure models under the Naïve-Bayes method. Recalling the Bayes formula in §3.1.2 in its joint probability form, we can define the joint probability of two proteins $\langle \mathbf{S},\mathbf{T} \rangle$ and their alignment $\mathcal{A}$ as follows:

$$\Pr(\mathcal{A},\, \langle \mathbf{S},\mathbf{T} \rangle) = \Pr(\mathcal{A}) \cdot \Pr(\langle \mathbf{S}_{\text{seq}}, \mathbf{T}_{\text{seq}} \rangle \mid \mathcal{A}) \cdot \Pr(\langle \mathbf{S}_{\text{struct}}, \mathbf{T}_{\text{struct}} \rangle \mid \mathcal{A})$$

This enables a consistent combination of the two information sources under a *unified alignment model*, with a *unified I-value* defined by:

$$I(\mathcal{A}, \langle \mathbf{S},\mathbf{T} \rangle) = \underbrace{I(\mathcal{A})}_{\text{First part}} + \underbrace{I(\langle \mathbf{S}_{\text{seq}}, \mathbf{T}_{\text{seq}} \rangle \,|\mathcal{A}) + I(\langle \mathbf{S}_{\text{struct}}, \mathbf{T}_{\text{struct}} \rangle \,|\mathcal{A})}_{\text{Second part}} \quad \text{bits} \tag{7.4}$$

Here, the first part accounts for the complexity of the alignment hypothesis $\mathcal{A}$ encoded as a three-state string using the adaptive coding method (Collier, 2016).

On the other hand, the second part accounts for the sum of independent second part message lengths resultant from Equation 4.1 (for sequence alignment) and Equation 7.2 (for structure alignment). For the sequence related second part message, the MMLSUM amino acid replacement matrix (inferred in Chapter 6) and UniProt proteome based amino acid null probabilities are applied on matches and insertions/deletions described by the alignment $\mathcal{A}$, respectively. For the structure related second part message, the C_α coordinates are encoded using the approach presented in Collier (2016).

Separately, the corresponding *unified null model* becomes:

$$I_{\text{NULL}}(\langle \mathbf{S}, \mathbf{T} \rangle) = I_{\text{NULL}}(\mathbf{S}_{\text{seq}}) + I_{\text{NULL}}(\mathbf{T}_{\text{seq}}) + I_{\text{NULL}}(\mathbf{S}_{\text{struct}}) + I_{\text{NULL}}(\mathbf{T}_{\text{struct}}) \quad \text{bits} \tag{7.5}$$

with *sequence null model* estimates inferred on UniProt sequences (see §4.4), and the *structure null model* described in Equation 7.1 to encode the C_α coordinates independently for each of the $\mathbf{S}$ and $\mathbf{T}$ proteins.

Accordingly, the test of significance for some alignment $\mathcal{A}$ follows the same routine of evaluating the compression gain ΔI, as the difference between the *unified null model* and the *unified alignment model*. If the difference is positive, the alignment is accepted, otherwise it is rejected.

Search for the optimal alignment: The above described *unified alignment model* is optimised under the objective function given by Equation 7.4 using the two-phase heuristic search procedure described by Collier (2016) summarised in the previous section. The alignment which gives the shortest message length for *unified alignment model*, yielding the most compression with respect to the *unified null model* message length (Equation 7.5), is chosen to be the optimal alignment under this proposed MML framework of unified sequence-structure alignment.

7.3 Results and Discussion

The following experiment establishes an initial attempt to validate the unified sequence-structure alignment framework.

7.3.1 Experimental Setting and Data

The performance of the unified framework was evaluated over a set of 1000 randomly sampled SCOP domain pairs across all SCOP classification levels: *family, superfamily, fold, class* and *decoy* (not belonging to the same class), reported in the thesis of (Collier, 2016).

Mainly, there are 200 pivot SCOP domains with which, 200 unique SCOP domains from each SCOP level are paired (i.e. a pivot domain P is paired with domain P_1 in the same family, domain P_2 in the same superfamily, domain P_3 in the same fold, domain P_4 in the same class, and domain P_5 in a different class). Optimal alignments were generated for all 1000 domain pairs under the MML structure alignment and MML sequence-structure alignment framework.

To understand this data from its sequence composition, we explore the *marginal probability models* for these 1000 protein domain pairs produced by the MML sequence alignment framework described in this thesis. The framework utilises MMLSUM Markov model of amino acid substitution in concert with the expected three-state machine parameter values under the optimal Dirichlet models derived in Chapter 6. The applied *null model* of the framework is the one inferred over UniProt protein sequences in §4.4.3. Figure 7.1 plots the amount of compression gain each *marginal probability model* has achieved with respect to their *sequence null model* (given in Equation 4.11) against their average evolutionary time t. The *family* level relationships yield an average compression of 27.7451 bits under a mean evolutionary time of $t = 226.3150$. The *superfamily* level relationships also display a positive average compression

Figure 7.1: Evolutionary time t versus compression (in bits) for 1000 SCOP domain pairs aligned under the marginal probability model of the MML based protein sequence alignment framework

of 9.7266 bits. This signifies the potential of *marginal probability model* to identify distant evolutionary relationships. The *fold* level nearly makes the cut with a 0.5513 average compression. The *class* and *decoy* levels both show a negative average compression of -0.1900 and -1.2253 bits. This is desirable since in general, these SCOP levels are considered to be not reflecting divergent evolutionary relationships.

7.3.2 Results for the Unified Sequence-Structure Alignment Framework

This section analyses how sequence-structure alignment differs from structure-only alignment (and clearly sequence-only alignment), by evaluating alignments for the above mentioned 1000 pairs of SCOP domains under the new unified framework, in comparison to those under the structure-only framework (Collier, 2016). Figure 7.2 and Figure 7.3 illustrate SCOP-level-wise box-plots of the compression gain/loss, alignment complexity ($I(\mathcal{A})$ in bits), the Shannon information content ($I(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$ in bits), RMSD (in Angstroms (Å)) and coverage (i.e. the number of matched amino acid pairs in the resultant alignment) of the 1000 alignments, separately for both frameworks.

Looking at the median compression gain for the SCOP *family* level domains, we observe an improvement of ~ 13.7 bits using the *unified* framework compared to *structure-only* framework. However, the *superfamily* and *fold* levels do not make much difference between the two alignment types (i.e. Only ~ 3 bits and ~ 1.6 bits difference in their median compression values, respectively). The *superfamily* level for unified alignment accounts for a median of 109.2295 bits, whereas that of structure only alignment produces a median of 112.2185 bits compression. For *class* and *decoy*, there is a significant difference with unified alignment giving a lesser compression than structure only alignment. It is favourable since they are known to be representing unrelated pairs of domains.

Alignment complexity distributions provide an interesting observation; where unified alignments bear more complexity than structure-only alignments on average, across *family*, *superfamily* and *fold* levels, it then drops for *fold* and *class* levels.

RMSD value distribution in Figure 7.3 favours unified alignment with improvements to RMSD values at *fold, class, decoy* level. This is explicable in terms of the coverage which has dropped compared to other levels. The *fold* and *class* level pairs are expected to bear no direct evolutionary relationship. However they can have similar local folding patterns which are likely to be picked up by the unified framework in terms of their amino acid sequence similarities. Consequently, matches made solely based on 3D coordinates could have been compensated for by unaligned regions, such that the coverage decreases and RMSD increases. The *superfamily* level statistics have not changed much. On the other hand, *family* level does not favour unified alignment in terms of RMSD, as the median and mean have gone up. This may be due to a trade-off between overall compression and RMSD.

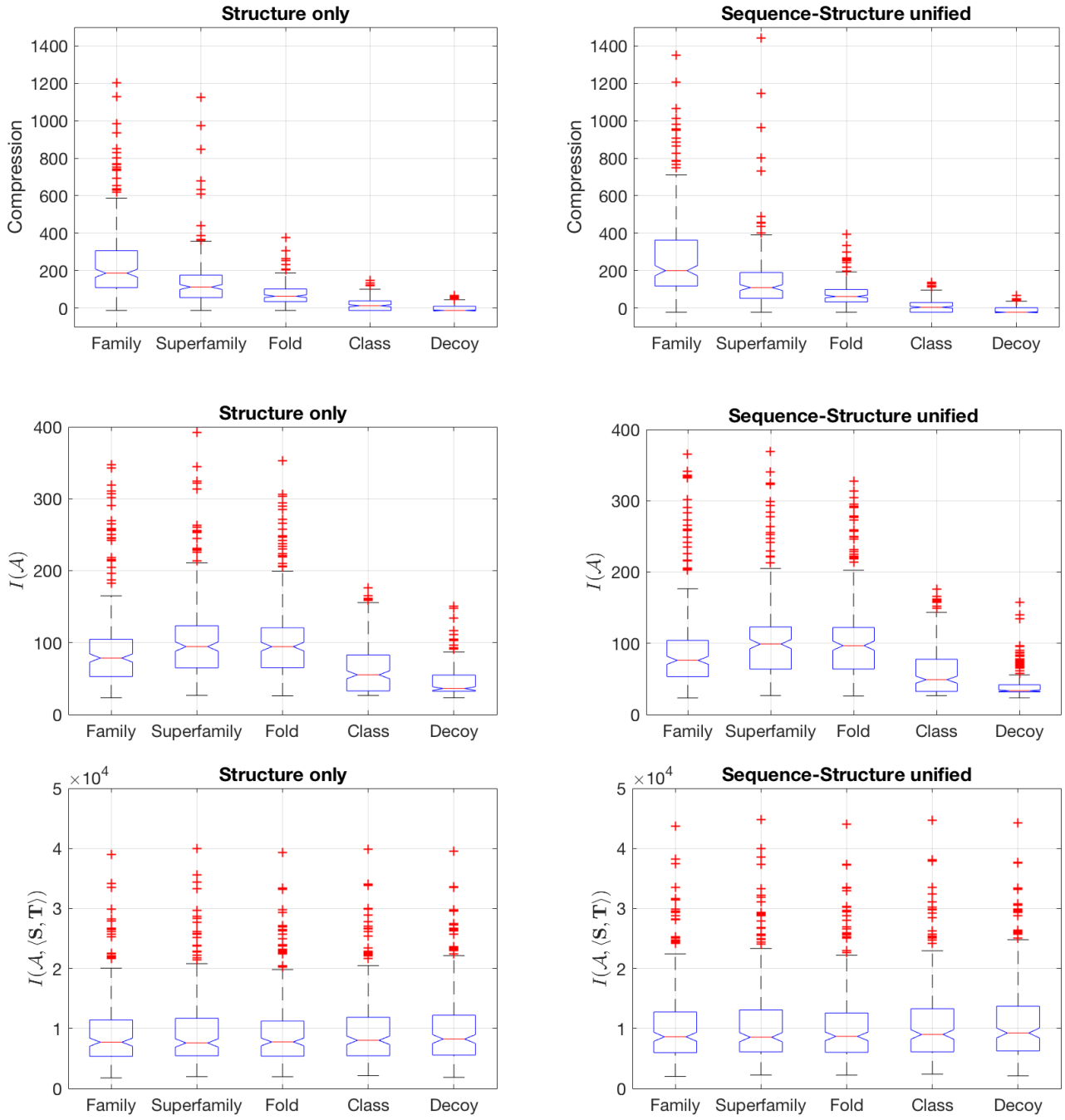


Figure 7.2: Results of structure-only alignments versus unified sequence-structure alignments under the MML framework. Top row compares the two in terms of compression (in bits) gained/lost with respect to their null models, shown as box-plots capturing the distribution of quartiles, for each SCOP-level. Similarly, the middle row compares their resultant alignment complexities, while the last row shows the distribution of their Shannon information content.

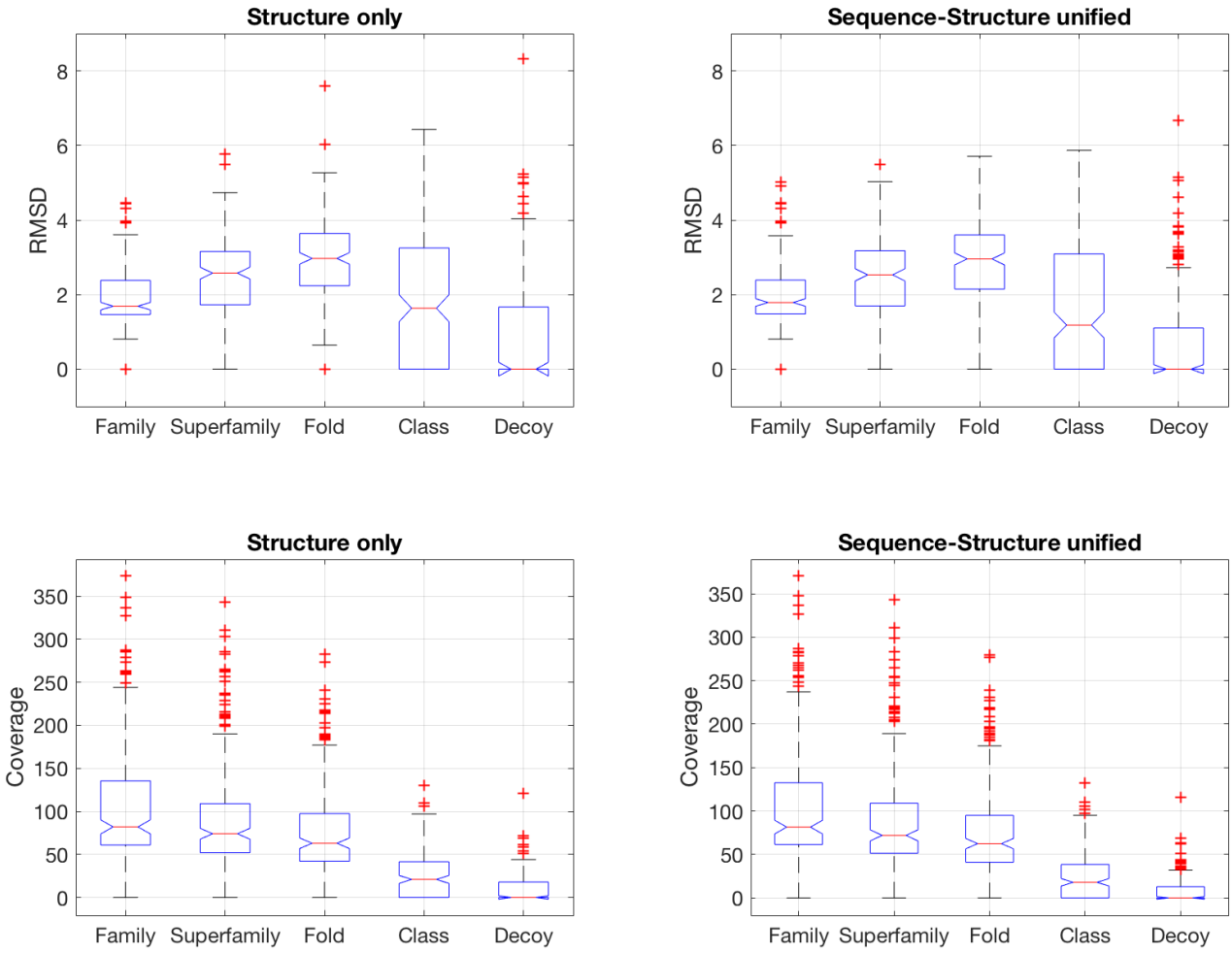


Figure 7.3: Comparison between structure-only and sequence-structure alignments in terms of their distributions of RMSD (top row) and coverage (bottom row), shown as box-plots for all 1000 domain pairs across the SCOP levels

7.3.3 A Case Study

As a case study, let us examine a randomly chosen domain-pair from our dataset that belongs to the same superfamily: `d1nara_` and `d1tvna`. SCOP organises this pair under the superfamily (Trans)glycosidases, a type of enzyme. The domain `d1nara_` is a seed storage protein in a plant species called *Vicia narbonensis* (known as the garden vetch). The domain `d1tvna` is a Endoglucanase protein found in a marine bacterium called *Pseudoalteromonas haloplanktis*. This section compares and contrasts the alignments of these highly-diverged proteins, generated using the unified and structure-only frameworks. Note that the sequence framework under the optimal alignment model infers their evolutionary time t under the MMLSUM substitution matrix as 599, whereas the model using marginal probability gives an average $t = 475$. The optimal alignment model does not yield positive compression, showing the difficulty of detecting their sequence similarity signal. However, the *marginal probability model* is able to detect that there exists some unspecified evolutionary relationship between them, with 11.7 bits of compression.

Figure 7.4 shows the similarities and differences between the optimal alignments produced by the MML structure-only and unified models. The reported unified alignment has a slightly better fit (RMSD 3.89 Å) compared to structure-only alignment (RMSD 4.08Å), although the unified framework's alignment yields a coverage of 205 correspondences (i.e. matched amino acids) compared to structure-only alignment coverage of 212 correspondences. Note, `d1nara_` and `d1tvna` have 289 and 293 amino acid residues in their chains, respectively.

Referring to their structure-only and sequence-structure alignments in Figure 7.4, only $\sim$ 57% of the aligned regions agree to the same substrings within their three-state alignment strings. The distinctions occur in the form alternative sequence matches and shifts in matches, due to unaligning certain parts (appearing as inserted or deleted stretches of amino acids). Figure 7.5 shows the subtle differences between the two types of alignments.

Focusing on the region highlighted by Figure 7.6, even though the superposed structural segments are arranged in the same way, a closer look reveals subtle variations in their spatial

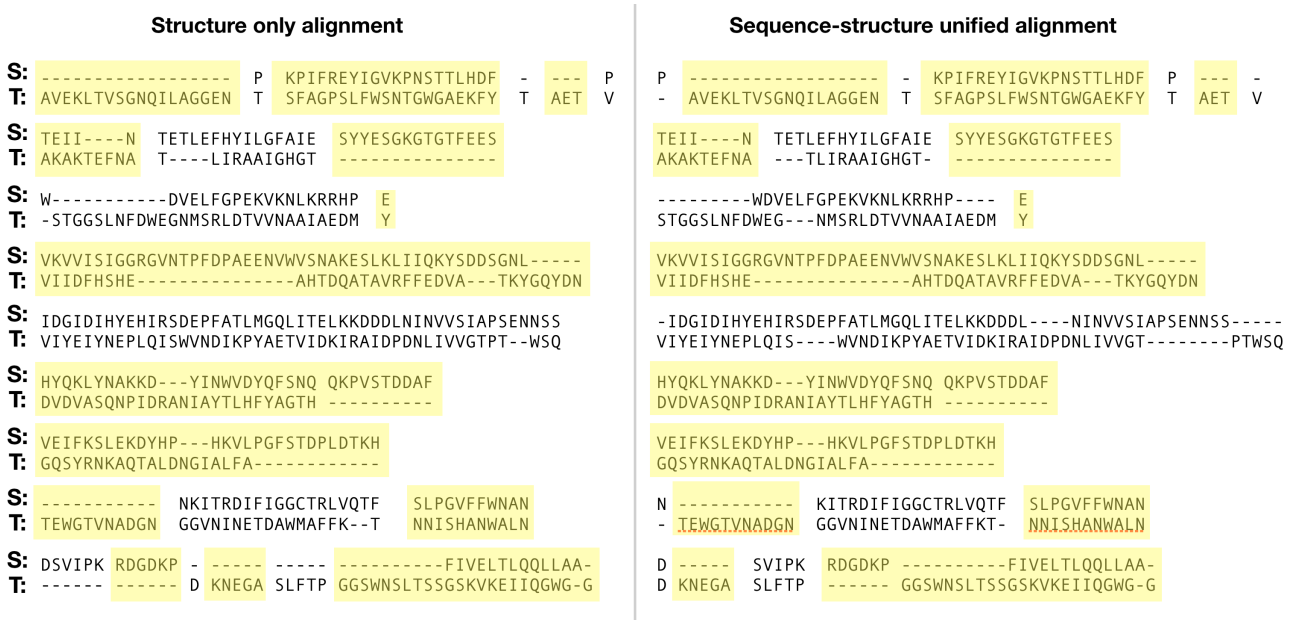
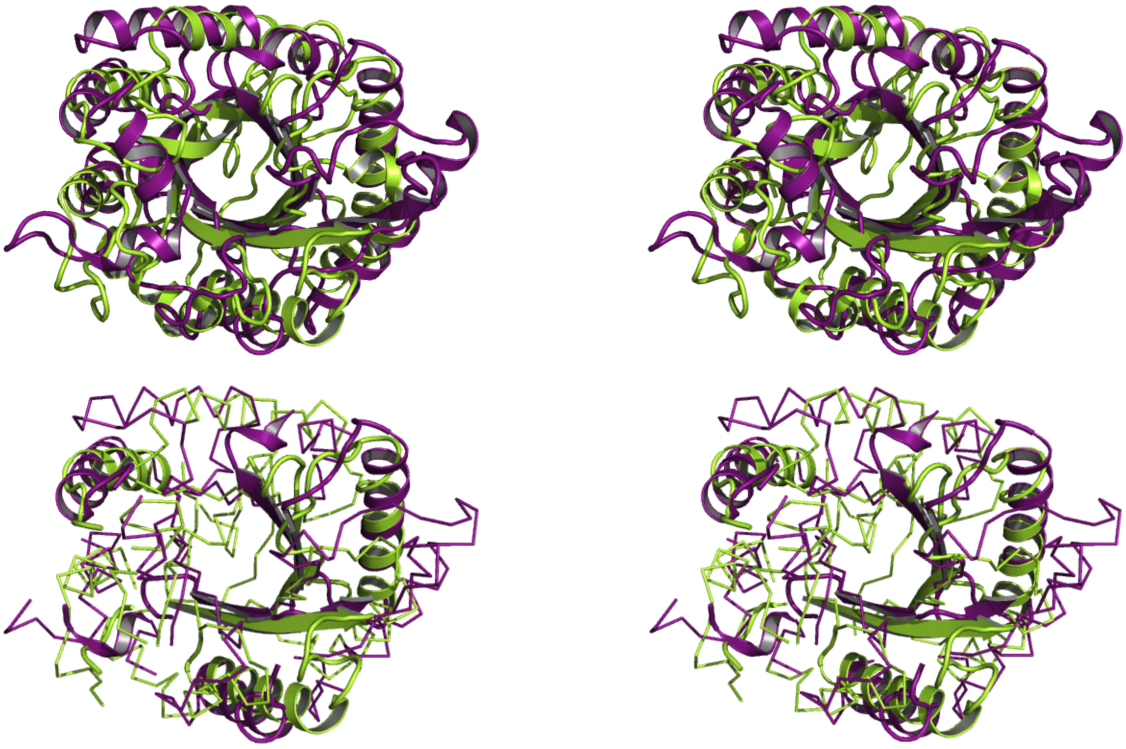


Figure 7.4: The similarities (yellow background) and differences (white background) between structure-only and unified sequence-structure alignments between the two SCOP domains: `d1nara_` (**S**) and `d1tvna` (**T**).



(a) (b)

Figure 7.5: Top row: superpositions of `d1nara_` (in purple colour) and `d1tvna` (in green colour) structures based on their optimal alignments produced by (a) structure-only, and (b) unified sequence-structure methods. Bottom row shows the same corresponding superpositions highlighting the regions of differences as thick ribbons, while the similarities are shown with thin lines. PyMOL (Schrödinger Inc., 2015) was used to render these molecular visualisations, and SST (Konagurthu et al., 2012) to delineate secondary structures (helices and strands of sheet).

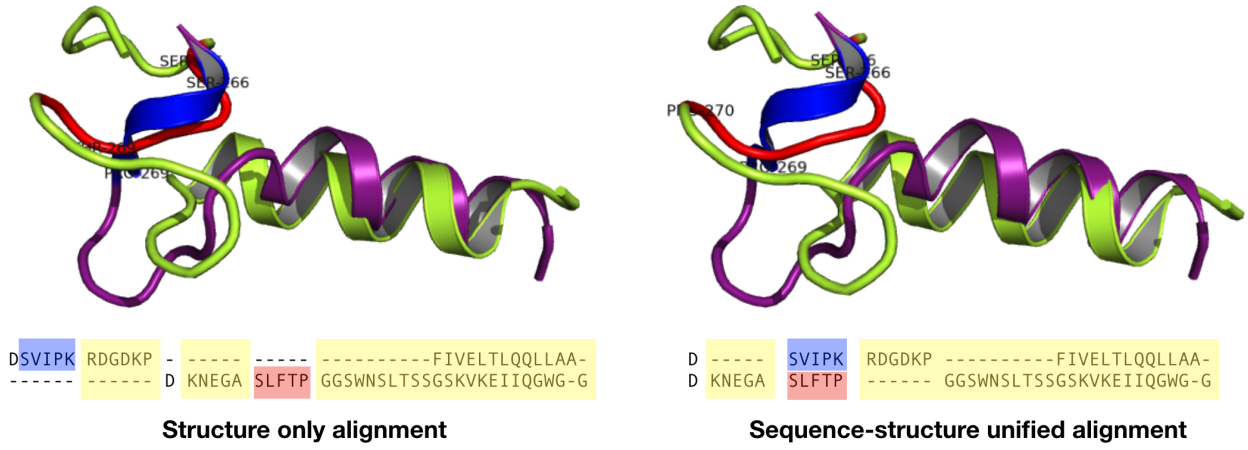


Figure 7.6: A region where alignments of the two SCOP domains: `d1nara_` and `d1tvna` differ. Blue highlighted amino acid stretch in `d1nara_` is matched with red highlighted amino acid stretch in `d1tvna` only by the unified alignment. Yellow highlighted are the regions that agree between the two alignments.

arrangements. The unified alignment has matched the `SLFTP` region in the domain `d1tvna` (highlighted in red) with `SVIPK` region in the domain `d1nara_` (highlighted in blue). The

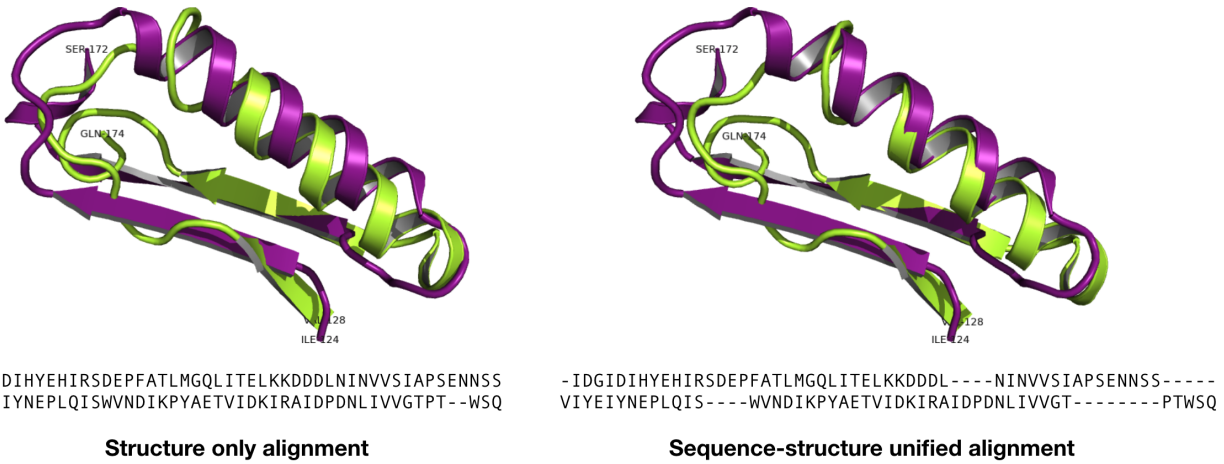


Figure 7.7: Another region where the optimal structure-only alignment and unified sequence-structure alignment of the two SCOP domains, `d1nara_` and `d1tvna`, differ. These regions within the two alignments have considerable differences in their patterns of matches and insertions/deletions.

identical match of `S` is not surprising. The match between `V` (Valine) and `L` (Leucine) makes sense since both are aliphatic residues. Next, the hydrophobic residue `I` (Isoleucine) can replace `F` (Phenylalanine), another hydrophobic residue. Replacement between `T` (Threonine) and `P` (Proline) is also sensible since they both fall under the group of small amino acid residues.

Another significantly different region of alignment is highlighted by Figure 7.7. While structure alignment gives a higher coverage, unified alignment introduces several inserted and deleted stretches of amino acids, resulting in a better fit between two alpha helices as noticeable from the visualisation. Such difference signifies the importance of combining both amino acid sequence and structure information during protein alignment.

The preliminary results and discussion presented in this chapter establishes a proof-of-concept for the potential in combining amino acid sequence and 3D structure information when aligning two proteins. The consistent unification of these two sources of information within the MML framework under a simple Naïve Bayes method opens up an avenue to investigate more on its strengths to identify evolutionary similarities between proteins.

Chapter 8

Conclusions and Future Directions

"...then there'll be no more work and there'll be perfect peace.
Really?
Yeah - that's Entropy, man!

– Flanders & Swann (in their song 'First And Second Law Of Thermodynamics'

This thesis revisits the classical problem of protein sequence alignment, and addresses it using the method of Minimum Message Length inference (Wallace, 2005) that links statistical inductive inference with lossless data compression. Using this information-theoretic framework, it learns a complete set of statistical models explaining amino acid substitutions, insertions and deletions. In revisiting this problem, this thesis attempts to continue the rigorous lines of research initiated by Allison and Wallace (1994); Allison et al. (1992b); Yee and Allison (1993) for DNA sequence alignment, and later by Eddy (1998) for protein alignment via pair Hidden Markov Models.

Specifically, this thesis formulates a Bayesian framework that permits an objective comparison of alignments between protein sequences in Shannon information terms. The framework directly addresses the necessary *complexity* versus *fit* trade-off when deciphering alignment relationships, and does so by learning parameters unsupervised and thereby avoiding *ad hoc* choices for them. An interesting feature of this framework is its ability to provide ways to explore closely-competing (alternative) alignments, especially over the marginal probability 'alignment landscapes' (Chapter 4). The thesis has also unified the statistical models explaining amino acid insertions and deletions, with those accounting amino acid substitutions (Chapters 5). Further, the thesis has examined a set of widely-known substitution models introduced in the past four decades. This is a revisit to substitution modelling for identifying strengths and weaknesses of existing models with the aim to improve them. It infers a new set of Markov models of protein evolution over several protein structural alignment benchmarks, with the purpose of improving the alignment framework's ability to detect significant sequence similarities (Chapter 6). A unique characteristic of this inference is that it incorporates both matches and gaps between protein sequences. This Markov model inference procedure is benchmark-agnostic and can be applied to learn average amino acid substitution patterns across any protein alignment benchmark. Finally, an effort is made to unify sequence information with structure information for generating sequence-structure alignments under the MML framework. For this, the MML sequence models introduced in this thesis are integrated with MML structure models introduced by the thesis of Collier (2016) (Chapter 7).

Overall, the results and insights presented here contribute to an improvement in solving the classical problem of protein sequence alignment, addressing key shortcomings and limitations

in the domain as highlighted in §2.5. As Habermann (2016) pointed out, presently, there does not exist a reliable statistical framework nor robust methods, especially for remote evolutionary relationship detection. Levy Karin et al. (2018) re-affirms that the field still has scope for growth. This thesis is therefore an attempt in that direction.

All models developed and implemented in this thesis follow a strictly probabilistic approach. The statistical framework of MML provides a consistent ground on which alignment models are evaluated. MML keeps the evaluation honest, with all parameters, their complexities and uncertainties inferred and handled explicitly. This includes the statement of all parameters and their precision, for example protein sequence and alignment lengths, evolutionary time parameter, three-state machine parameters etc. This consideration is important to address any inference problem. The presented framework is extendable, where the models across multiple sources of information can be integrated with existing models due to the additive property of information.

To conclude, a couple of future directions of research building on this thesis are briefly discussed below.

Multiple Protein Sequence Alignment

Quoting Arthur Lesk "*two homologous sequences whisper...multiple alignment shouts out loud*" (Gusfield, 1997). Thus a major direction would be an extension of the MML framework to facilitate multiple sequence alignment. This requires considerable thought and effort, even when using the common progressive alignment approach of Feng and Doolittle (1987). However, a good starting point is the previous work of Allison et al. (1992a) addressing the phylogenetic tree inference and computation of multiple alignments using MML. Many considerations will have to be extended for this problem. For example, the relationship is no longer a single three-state alignment string. Instead, It can be a collection of three-state alignment strings with respect to one chosen sequence in the set of multiple sequences for which we aim to find an optimal alignment. A suitable method is to formulate this as an evolutionary tree hypothesis, in which case the tree topology, all parameters and their associated complexities have to be concretely defined and inferred under the MML criterion.

Exploring the Patterns of Conservation of Sequence and Structure in Evolution

Better understanding of the relationship between sequence conservation and structure conservation is an direction of future exploration using the MML models handling sequence, structure and sequence-structure source of information. Thus, it will be useful to explore quantitatively and qualitatively the differences between sequence alignment, structure alignment and sequence-structure alignment, and study their consensus (in terms of shared correspondences) and identify the point where they deviate from each other. Mainly, this will extend the preliminary exercise discussed in Chapter 7. Further, the MML sequence alignment framework can be applied to infer a dictionary of sequence-structure determinants (Lesk et al., 2018).

Appendix A

Information on Proteome Data Sources

Table A.1: Information on proteomes across Eukaryota, Bacteria and Archea

Organsim	**Uniprot ID**	**Description**
Escherichia coli (strain K12)	UP000000625	Serves for suppressing harmful bacterial growth and synthesising vitamins in the body
Homo sapiens	UP000005640	This is us, the humans
Arabidopsis thaliana	UP000006548	A flowering plant; the first plant genome to be sequenced
Mus musculus	UP000000589	House rat, the second mammal genome to be sequenced
Drosophila melanogaster	UP000000803	Fruit fly
Saccharomyces cerevisiae (S288c)	UP000002311	Unicellular Fungus
Methanocaldococcus jannaschii	UP000000805	Strict anaerobic organism found in deep sea
Plasmodium ovale wallikeri	UP000078550	Causes tertian Malaria
Clostridium tetani	UP000001412	A soil bacterium that causes Tetanus; It has an AT rich genome
Mycobacterium tuberculosis	UP000001584	A pathogenic bacterium that causes Tuberculosis; It has a GC rich genome
Bacillus subtilis (strain 168)	UP000001570	Harmless bacterium found in soil
Lactococcus lactis (strain IL1403)	UP000002196	A bacterium used in milk industry for cheese production; It has an AT rich genome
Streptomyces coelicolor	UP000001973	A bacterium with an ability to produce metabolites including antibiotics; It has a GC rich genome

*based on UniProt (UniProt Consortium and others, 2017) and Yu et al. (2003)

Table A.2: Information on viral proteomes

Organsim	**Uniprot ID**
Paramecium bursaria Chlorella virus 1 (PBCV-1)	UP000000862
Bacillus virus G	UP000009273
Cafeteria roenbergensis virus (strain BV-PW1 (CroV))	UP000029781
Emiliania huxleyi virus (EhV-86)	UP000000863
Human SARS coronavirus (SARS CoV)	UP000000354

Appendix B

Amino Acid Residue Clusters with tSNE Plots

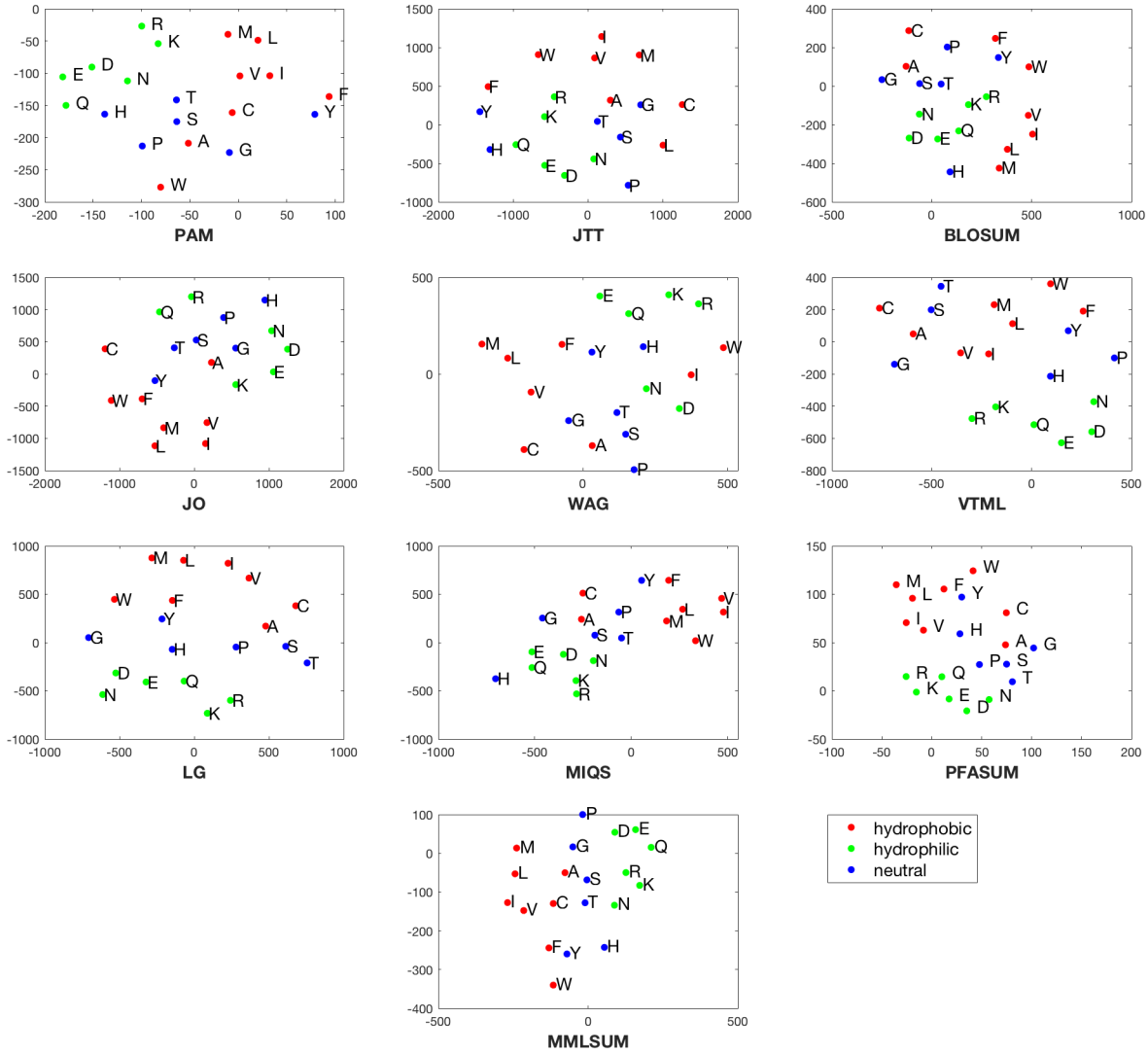


Figure B.1: tSNE plots of different substitution matrices reflecting amino acid clusters based on their hydropathy as classified by Lefranc et al. (2015)

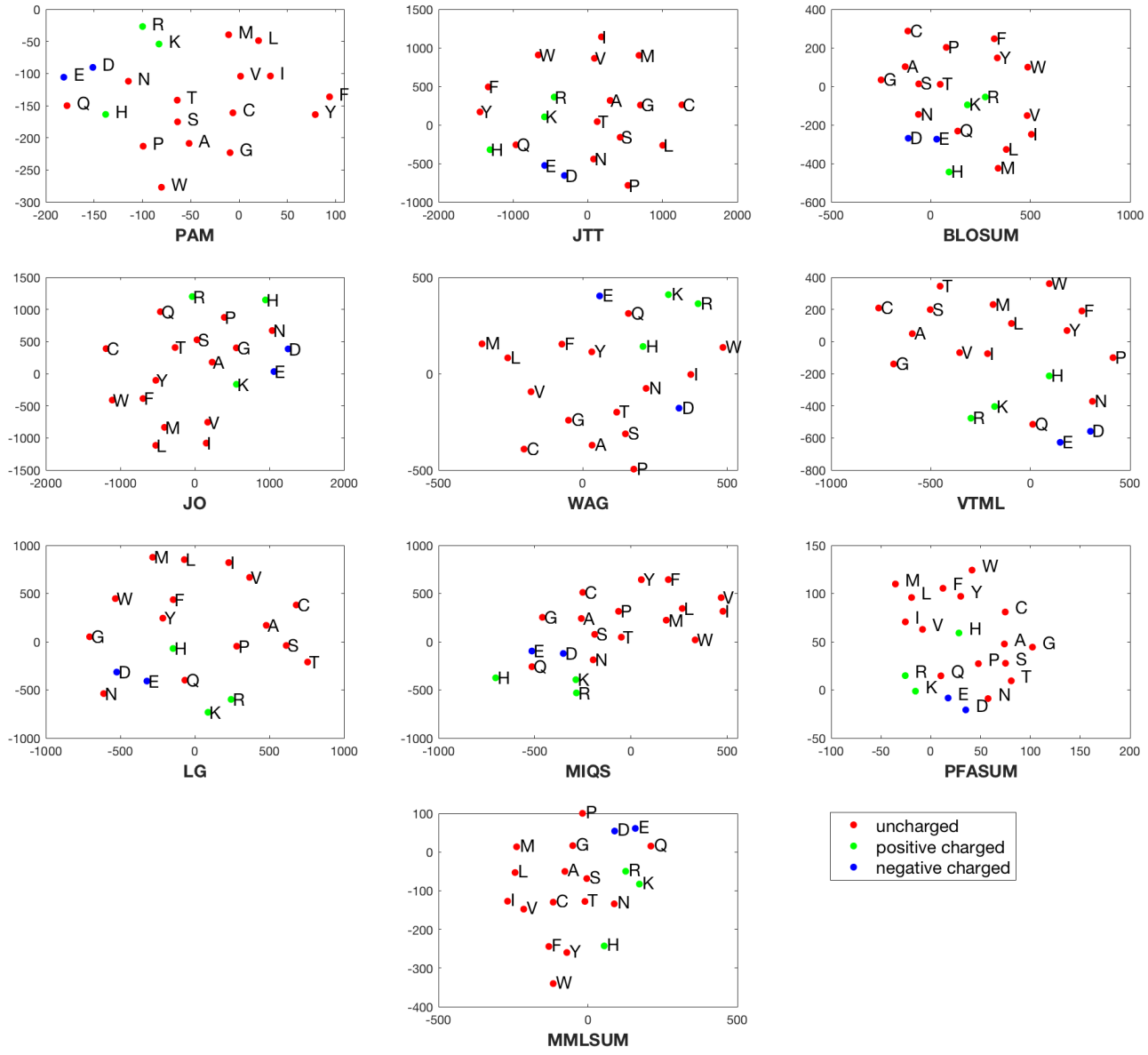


Figure B.2: tSNE plots of different substitution matrices reflecting amino acid clusters based on their charge as classified by Lefranc et al. (2015)

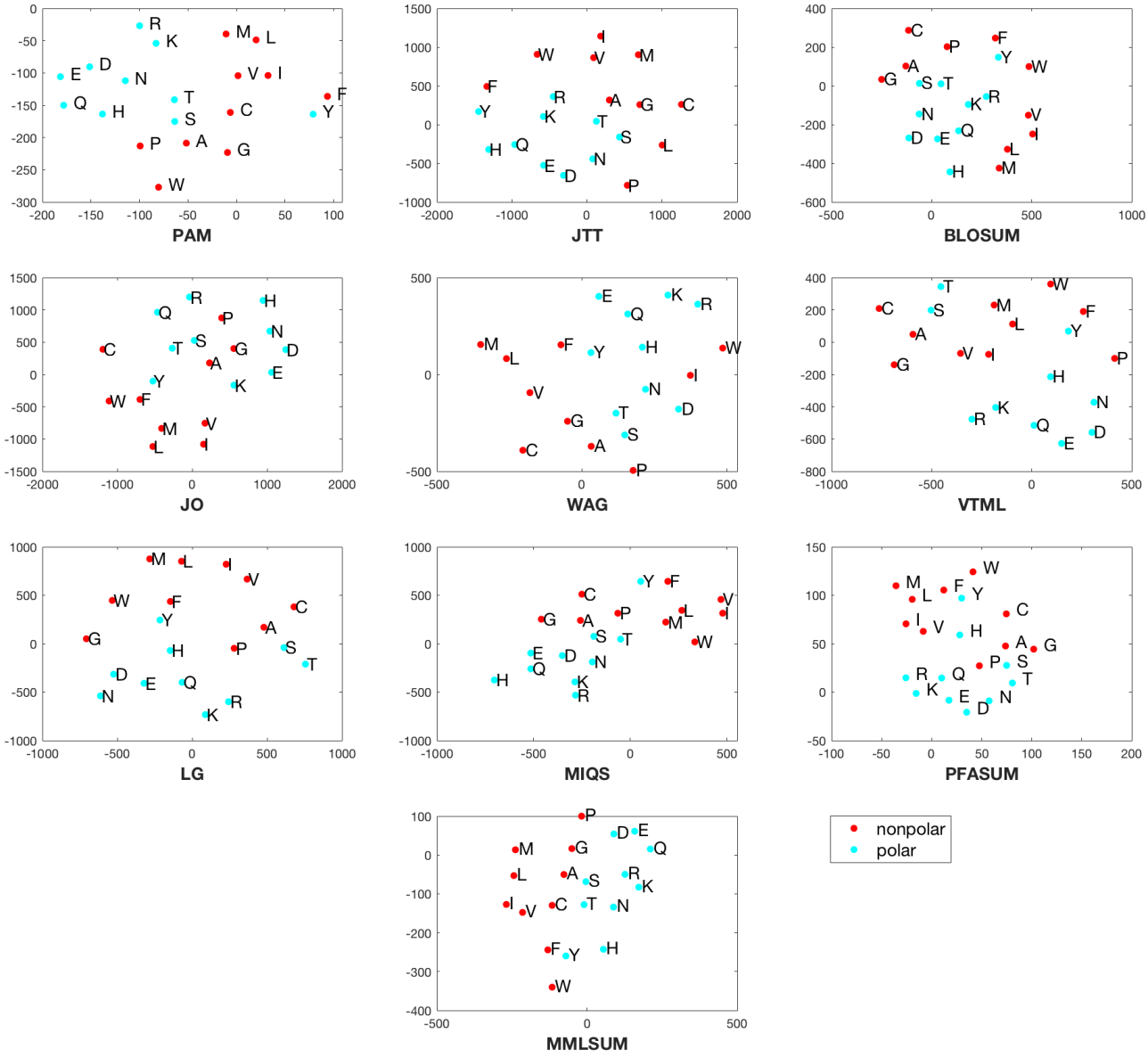


Figure B.3: tSNE plots of different substitution matrices reflecting amino acid clusters based on their polarity as classified by Lefranc et al. (2015)

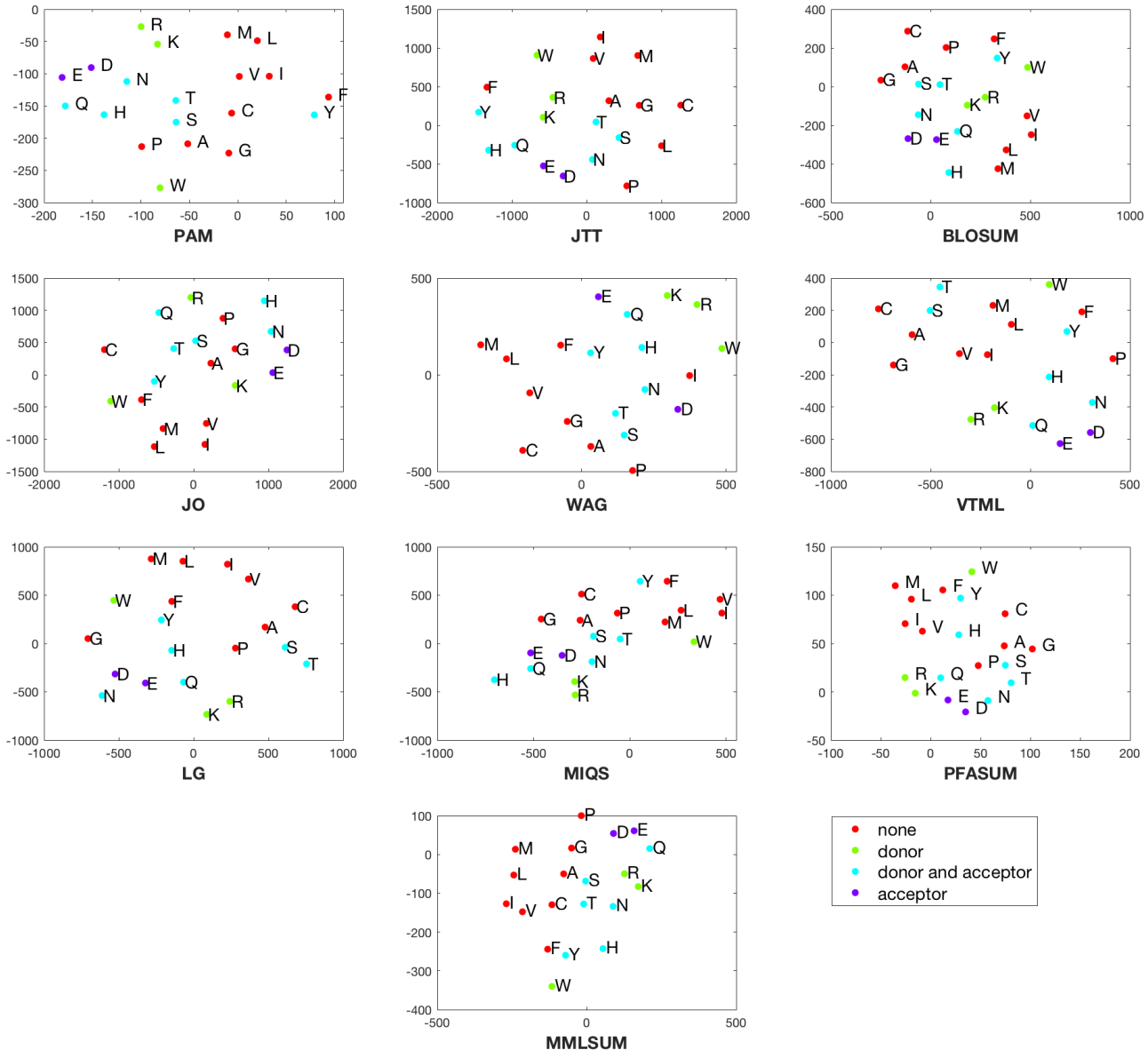


Figure B.4: tSNE plots of different substitution matrices reflecting amino acid clusters based on their hydrogen donor or acceptor status as classified by Lefranc et al. (2015)

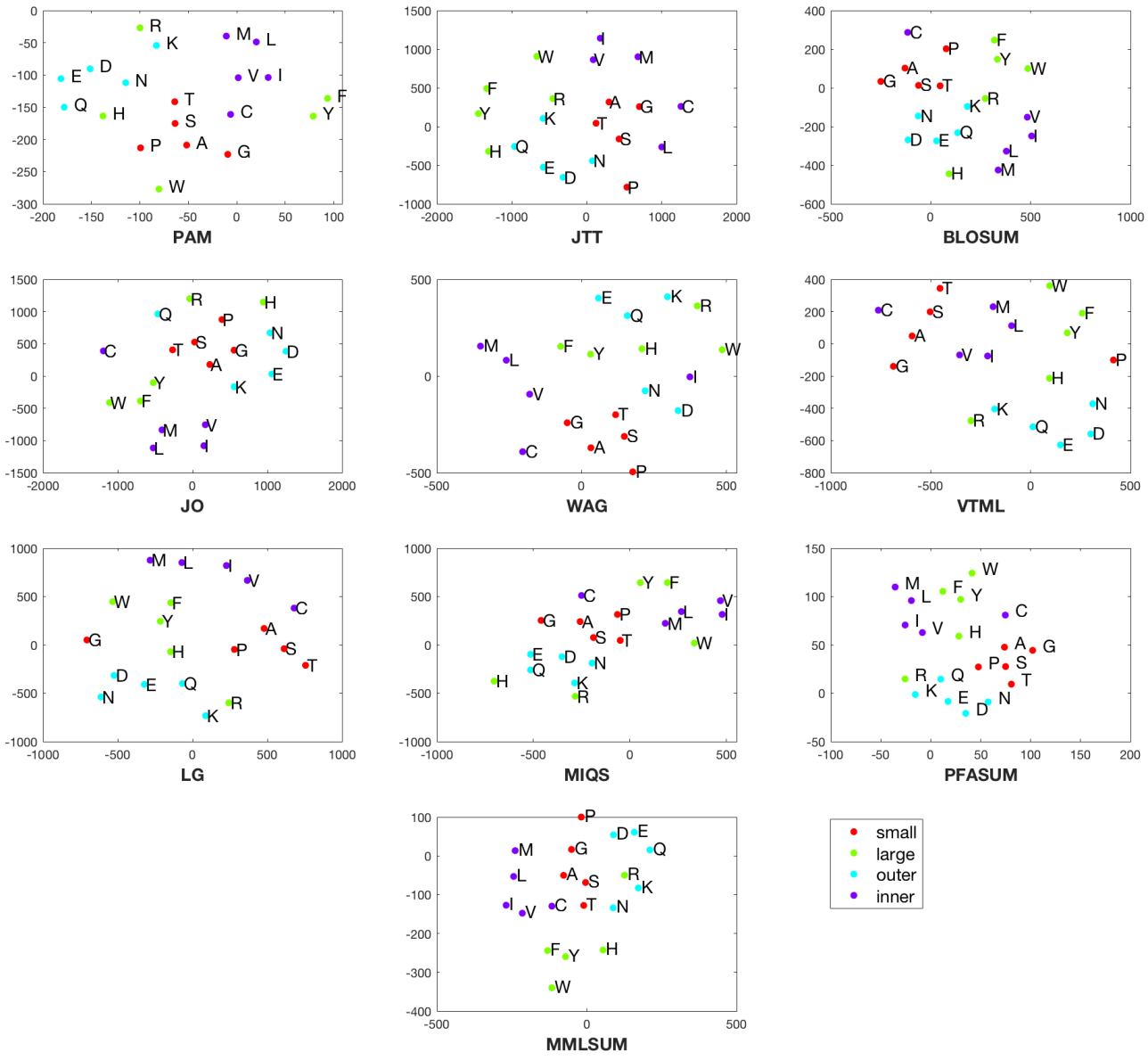


Figure B.5: tSNE plots of different substitution matrices reflecting amino acid clusters based on their volume and exposure as classified by Swanson (1984)

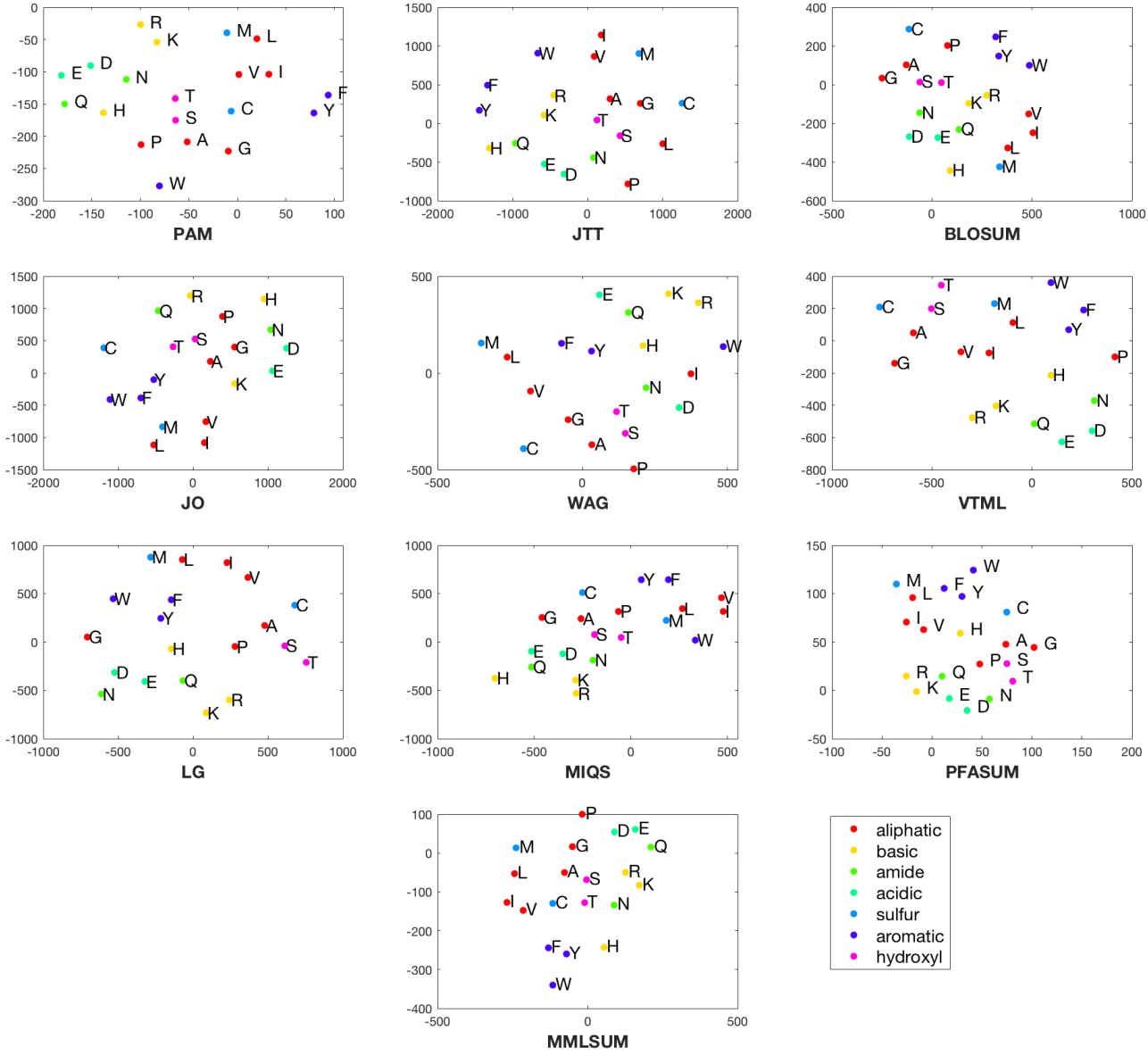


Figure B.6: tSNE plots of different substitution matrices reflecting amino acid clusters based on their chemical properties as classified by Lefranc et al. (2015)

References

Adachi, J. and Hasegawa, M. (1996). *MOLPHY version 2.3: programs for molecular phylogenetics based on maximum likelihood*, number 28, Institute of Statistical Mathematics Tokyo.

Adachi, J., Waddell, P. J., Martin, W. and Hasegawa, M. (2000). Plastid genome phylogeny and a model of amino acid substitution for proteins encoded by chloroplast DNA, *Journal of Molecular Evolution* **50**(4): 348–358.

Ahrens, J. H. and Dieter, U. (1982). Generating gamma variates by a modified rejection technique, *Communications of the ACM* **25**(1): 47–54.

Al Ait, L., Yamak, Z. and Morgenstern, B. (2013). DIALIGN at GOBICS—multiple sequence alignment using various sources of external information, *Nucleic Acids Research* **41**(W1): W3–W7.

Allison, L. (1993). Normalization of affine gap costs used in optimal sequence alignment, *Journal of Theoretical Biology* **161**(2): 263–269.
URL: *http://dx.doi.org/10.1006/jtbi.1993.1054*

Allison, L. (2018). *Coding Ockham's Razor*, Springer.
URL: *https://doi.org/10.1007/978-3-319-76433-7*

Allison, L., Konagurthu, A. and Schmidt, D. (2019). On universal codes for integers: Wallace tree, Elias omega and variations, *arXiv preprint arXiv:1906.05004* .

Allison, L. and Wallace, C. (1994). The posterior probability distribution of alignments and its application to parameter estimation of evolutionary trees and to optimization of multiple alignments, *Journal of Molecular Evolution* **39**(4): 418–430.

Allison, L., Wallace, C. S. and Yee, C. N. (1992a). Minimum message length encoding, evolutionary trees and multiple-alignment, *25th Hawaii Int. Conf. on System Sci.* (TR 91 155, Monash University, Department of Computer Science, 1991).

Allison, L., Wallace, C. and Yee, C. (1992b). Finite-state models in the alignment of macromolecules, *Journal of Molecular Evolution* **35**(1): 77–89.

Altschul, S. F. (1993). A protein alignment scoring system sensitive at all evolutionary distances, *Journal of Molecular Evolution* **36**(3): 290–300.

Altschul, S. F., Bundschuh, R., Olsen, R. and Hwa, T. (2001). The estimation of statistical parameters for local alignment score distributions, *Nucleic Acids Research* **29**(2): 351–361.

Altschul, S. F., Gish, W., Miller, W., Myers, E. W. and Lipman, D. J. (1990). Basic local alignment search tool, *Journal of Molecular Biology* **215**(3): 403–410.

Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W. and Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs, *Nucleic Acids Research* **25**(17): 3389–3402.

Andreeva, A., Kulesha, E., Gough, J. and Murzin, A. G. (2020). The SCOP database in 2020: expanded classification of representative family and superfamily domains of known protein structures, *Nucleic Acids Research* **48**(D1): D376–D382.

Apweiler, R., Attwood, T. K., Bairoch, A., Bateman, A., Birney, E., Biswas, M., Bucher, P., Cerutti, L., Corpet, F., Croning, M. D. et al. (2001). The InterPro database, an integrated documentation resource for protein families, domains and functional sites, *Nucleic Acids Research* **29**(1): 37–40.

Arvestad, L. (2006). Efficient methods for estimating amino acid replacement rates, *Journal of Molecular Evolution* **62**(6): 663–673.

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T. et al. (2000). Gene Ontology: tool for the unification of biology, *Nature Genetics* **25**(1): 25–29.

Attard, P. (2002). *Thermodynamics and Statistical Mechanics: Equilibrium by Entropy Maximisation*, Elsevier.

Bairoch, A. and Bucher, P. (1994). PROSITE: recent developments., *Nucleic Acids Research* **22**(17): 3583.

Barker, W. C., Garavelli, J. S., Huang, H., McGarvey, P. B., Orcutt, B. C., Srinivasarao, G. Y., Xiao, C., Yeh, L.-S. L., Ledley, R. S., Janda, J. F. et al. (2000). The protein information resource (PIR), *Nucleic Acids Research* **28**(1): 41–44.

Barrientos, A. (2003). Yeast models of human mitochondrial diseases, *IUBMB Life* **55**(2): 83–95.

Barton, G. J. and Sternberg, M. J. (1987). Evaluation and improvements in the automatic alignment of protein sequences, *Protein Engineering, Design and Selection* **1**(2): 89–94.

Bastien, O. and Maréchal, E. (2008). Evolution of biological sequences implies an extreme value distribution of type i for both global and local pairwise alignment scores, *BMC Bioinformatics* **9**(1): 332.

Bateman, A., Birney, E., Durbin, R., Eddy, S. R., Howe, K. L. and Sonnhammer, E. L. (2000). The Pfam protein families database, *Nucleic Acids Research* **28**(1): 263–266.

Bayes, T. (1763). LII. An essay towards solving a problem in the doctrine of chances. by the late Rev. Mr. Bayes, FRS communicated by Mr. Price, in a letter to John Canton, AMFR S, *Philosophical Transactions of the Royal Society of London* (53): 370–418.
URL: *https://www.jstor.org/stable/2333180*

Bellman, R. et al. (1954). The theory of dynamic programming, *Bulletin of the American Mathematical Society* **60**(6): 503–515.

Benner, S. A., Cohen, M. A. and Gonnet, G. H. (1993). Empirical and structural models for insertions and deletions in the divergent evolution of proteins, *Journal of Molecular Biology* **229**(4): 1065–1082.

Bennet, S., Cohen, M. A. and Gonnet, G. H. (1994). Amino acid substitution during functionally constrained divergent evolution of protein sequences, *Protein Engineering, Design and Selection* **7**(11): 1323–1332.

Berestycki, N. (2016). Mixing times of Markov chains: Techniques and examples, *Alea-Latin American Journal of Probability and Mathematical Statistics* .
URL: *http://www.statslab.cam.ac.uk/~beresty/teach/Mixing/mixing3.pdf*

Berman, H., Henrick, K. and Nakamura, H. (2003). Announcing the worldwide protein data bank, *Nature Structural & Molecular Biology* **10**(12): 980–980.

Blake, J. D. and Cohen, F. E. (2001). Pairwise sequence alignment below the twilight zone, *Journal of Molecular Biology* **307**(2): 721–735.

Boutonnet, N. S., Rooman, M. J., Ochagavia, M.-E., Richelle, J. and Wodak, S. J. (1995). Optimal protein structure alignments by multiple linkage clustering: application to distantly related proteins, *Protein Engineering, Design and Selection* **8**(7): 647–662.

Brown, D. G. and Truszkowski, J. (2012). Fast phylogenetic tree reconstruction using locality-sensitive hashing, *International Workshop on Algorithms in Bioinformatics*, Springer, pp. 14–29.

Bryant, S. H. and Altschul, S. F. (1995). Statistics of sequence-structure threading, *Current Opinion in Structural Biology* **5**(2): 236–244.

Buchan, D. W. and Jones, D. T. (2017). EigenTHREADER: analogous protein fold recognition by efficient contact map threading, *Bioinformatics* **33**(17): 2684–2690.

Cao, L., Goreshnik, I., Coventry, B., Case, J. B., Miller, L., Kozodoy, L., Chen, R. E., Carter, L., Walls, A. C., Park, Y.-J. et al. (2020). De novo design of picomolar SARS-CoV-2 miniprotein inhibitors, *Science* **370**(6515): 426–431.

Cartwright, R. A. (2006). Logarithmic gap costs decrease alignment accuracy, *BMC Bioinformatics* **7**(1): 527.

Chang, M. S. and Benner, S. A. (2004). Empirical analysis of protein insertions and deletions determining parameters for the correct placement of gaps in protein sequence alignments, *Journal of Molecular Biology* **341**(2): 617–631.

Cheatle Jarvela, A. M., Brubaker, L., Vedenko, A., Gupta, A., Armitage, B. A., Bulyk, M. L. and Hinman, V. F. (2014). Modular evolution of DNA-binding preference of a tbrain transcription factor provides a mechanism for modifying gene regulatory networks, *Molecular Biology and Evolution* **31**(10): 2672–2688.

Chothia, C., Gough, J., Vogel, C. and Teichmann, S. A. (2003). Evolution of the protein repertoire, *Science* **300**(5626): 1701–1703.

Chothia, C. and Lesk, A. M. (1986). The relation between the divergence of sequence and structure in proteins., *The EMBO Journal* **5**(4): 823.

Cleary, J. and Witten, I. (1984). Data compression using adaptive coding and partial string matching, *IEEE Transactions on Communications* **32**(4): 396–402.

Collier, J. (2016). *Statistical Inductive Inference of Protein Structural Alignments*, PhD thesis, Monash University.
URL: https://doi.org/10.4225/03/58b79813d9110

Collier, J. H., Allison, L., Lesk, A. M., Garcia de la Banda, M. and Konagurthu, A. S. (2014). A new statistical framework to assess structural alignment quality using information compression, *Bioinformatics* **30**(17): i512–i518.
URL: https://doi.org/10.1093/bioinformatics/btu460

Collier, J. H., Allison, L., Lesk, A. M., Stuckey, P. J., Garcia de la Banda, M. and Konagurthu, A. S. (2017). Statistical inference of protein structural alignments using information and compression, *Bioinformatics* **33**(7): 1005–1013.
URL: https://doi.org/10.1093/bioinformatics/btw757

Conway, J. H. and Sloane, N. J. (1984). On the Voronoi regions of certain lattices, *SIAM Journal on Algebraic Discrete Methods* **5**(3): 294–305.

Dang, C. C., Le, Q. S., Gascuel, O. and Le, V. S. (2010). FLU, an amino acid substitution model for influenza proteins, *BMC Evolutionary Biology* **10**(1): 99.

Daniels, N., Kumar, A., Cowen, L. and Menke, M. (2011). Touring protein space with matt, *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **9**(1): 286–293.

Darwin, C. (1859). *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*, W. Clowes & Sons.

Dayhoff, M. (1978). Survey of new data and computer methods of analysis, *Atlas of Protein Sequence and Structure* .

Dayhoff, M. O., Eck, R. V., Chang, M. A. and Sochard, M. R. (1965). Atlas of Protein Sequence and Structure.
URL: https://ntrs.nasa.gov/api/citations/19660014530/downloads/19660014530.pdf

Dayhoff, M., Schwartz, R. and Orcutt, B. (1978). A model of evolutionary change in proteins, *Atlas of Protein Sequence and Structure*, Vol. 5, National Biomedical Research Foundation Silver Spring, MD, pp. 345–352.

Devauchelle, C., Grossmann, A., Hénaut, A., Holschneider, M., Monnerot, M., Risler, J.-L. and Torrésani, B. (2001). Rate matrices for analyzing large families of protein sequences, *Journal of Computational Biology* **8**(4): 381–399.

Do, C. B., Gross, S. S. and Batzoglou, S. (2006). CONTRAlign: discriminative training for protein sequence alignment, *Annual International Conference on Research in Computational Molecular Biology*, Springer, pp. 160–174.

Do, C. B., Mahabhashyam, M. S., Brudno, M. and Batzoglou, S. (2005). ProbCons: Probabilistic consistency-based multiple sequence alignment, *Genome Research* **15**(2): 330–340.

Doolittle, R. F. (1986). *Of URFs and ORFs: A Primer on how to Analyze Derived Amino Acid Sequences*, University Science Books.

Dosztanyi, Z. and Torda, A. E. (2001). Amino acid similarity matrices based on force fields, *Bioinformatics* **17**(8): 686–699.

Douzery, E. J., Snell, E. A., Bapteste, E., Delsuc, F. and Philippe, H. (2004). The timing of eukaryotic evolution: does a relaxed molecular clock reconcile proteins and fossils?, *Proceedings of the National Academy of Sciences* **101**(43): 15386–15391.

Durand, D. (2015). Lecture: Amino Acid Substitution Matrices – BLOSUM matrices.
URL: *http://www.cs.cmu.edu/~durand/03-711/2015/Lectures/Blosum-2015.pdf*

Durbin, R., Eddy, S. R., Krogh, A. and Mitchison, G. (1998). *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*, Cambridge University Press.

Eddy, S. R. (1998). Profile hidden Markov models., *Bioinformatics* **14**(9): 755–763.

Edgar, R. (2004). MUSCLE: multiple sequence alignment with high accuracy and high throughput, *Nucleic Acids Research* **32**(5): 1792–1797.

Edman, P. et al. (1950). Method for determination of the amino acid sequence in peptides, *Acta Chemica Scandinavica* **4**(7): 283–293.

El-Gebali, S., Mistry, J., Bateman, A., Eddy, S. R., Luciani, A., Potter, S. C., Qureshi, M., Richardson, L. J., Salazar, G. A., Smart, A. et al. (2019). The Pfam protein families database in 2019, *Nucleic Acids Research* **47**(D1): D427–D432.

Elias, P. (1975). Universal codeword sets and representations of the integers, *IEEE Transactions on Information Theory* **21**(2): 194–203.

Fallaize, C. J., Green, P. J., Mardia, K. V. and Barber, S. (2020). Bayesian protein sequence and structure alignment, *Journal of the Royal Statistical Society: Series C (Applied Statistics)* .

Farheen, N., Sen, N., Nair, S., Tan, K. P. and Madhusudhan, M. (2017). Depth dependent amino acid substitution matrices and their use in predicting deleterious mutations, *Progress in Biophysics and Molecular Biology* **128**: 14–23.

Felsenstein, J. (1973). Maximum likelihood and minimum-steps methods for estimating evolutionary trees from data on discrete characters, *Systematic Biology* **22**(3): 240–249.

Feng, D.-F. and Doolittle, R. F. (1987). Progressive sequence alignment as a prerequisitetto correct phylogenetic trees, *Journal of Molecular Evolution* **25**(4): 351–360.

Feng, D.-F. and Doolittle, R. F. (1990). Progressive alignment and phylogenetic tree construction of protein sequences, *Methods in Enzymology* **183**: 375–387.

Feng, D., Johnson, M. and Doolittle, R. (1985). Aligning amino acid sequences: comparison of commonly used methods, *Journal of Molecular Evolution* **21**(2): 112–125.

Fischer, M. G., Allen, M. J., Wilson, W. H. and Suttle, C. A. (2010). Giant virus with a remarkable complement of genes infects marine zooplankton, *Proceedings of the National Academy of Sciences* **107**(45): 19508–19513.

Fitch, W. M. (1966). An improved method of testing for evolutionary homology, *Journal of Molecular Biology* **16**(1): 9–16.

Fitch, W. M. (1971). Toward defining the course of evolution: minimum change for a specific tree topology, *Systematic Biology* **20**(4): 406–416.

Fitch, W. M. and Smith, T. F. (1983). Optimal sequence alignments, *Proceedings of the National Academy of Sciences* **80**(5): 1382–1386.

French, S. and Robson, B. (1983). What is a conservative substitution?, *Journal of Molecular Evolution* **19**(2): 171–175.

Frenkel, D., Schrenk, K. J. and Martiniani, S. (2017). Monte Carlo sampling for stochastic weight functions, *Proceedings of the National Academy of Sciences* **114**(27): 6924–6929.

Frith, M. C. (2020). How sequence alignment scores correspond to probability models, *Bioinformatics* **36**(2): 408–415.

Frobenius, G., Frobenius, F. G., Frobenius, F. G., Frobenius, F. G. and Mathematician, G. (1912). Über matrizen aus nicht negativen elementen.

Garrett, R. A. (1996). Genomes: Methanococcus jannaschii and the golden fleece, *Current Biology* **6**(11): 1377–1380.

Gene Ontology Consortium (2004). The Gene Ontology (GO) database and informatics resource, *Nucleic Acids Research* **32**(suppl_1): D258–D261.

George, D. G., Barker, W. C. and Hunt, L. T. (1990). Mutation data matrix and its uses, *Methods in Enzymology* **183**: 333–351.

Gonnet, G. H., Cohen, M. A. and Benner, S. A. (1992). Exhaustive matching of the entire protein sequence database, *Science* **256**(5062): 1443–1445.

Gonnet, G. and Korostensky, C. (1999). Optimal scoring matrices for estimating distances between aligned sequences, *Technical report*, ETH Zurich.

Gotoh, O. (1982). An improved algorithm for matching biological sequences, *Journal of Molecular Biology* **162**(3): 705–708.

Grantham, R. (1974). Amino acid difference formula to help explain protein evolution, *Science* **185**(4154): 862–864.

Gui, M., Song, W., Zhou, H., Xu, J., Chen, S., Xiang, Y. and Wang, X. (2017). Cryo-electron microscopy structures of the SARS-CoV spike glycoprotein reveal a prerequisite conformational state for receptor binding, *Cell Research* **27**(1): 119–129.

Gusfield, D. (1997). *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*, Cambridge University Press.

Habermann, B. H. (2016). Oh Brother, Where Art Thou? Finding orthologs in the twilight and midnight zones of sequence similarity, *Evolutionary Biology*, Springer, pp. 393–419.

Hadley, C. and Jones, D. T. (1999). A systematic comparison of protein structure classifications: SCOP, CATH and FSSP, *Structure* **7**(9): 1099–1112.

Hasegawa, H. and Holm, L. (2009). Advances and pitfalls of protein structural alignment, *Current Opinion in Structural Biology* **19**(3): 341–348.

Henikoff, S. and Henikoff, J. G. (1992). Amino acid substitution matrices from protein blocks, *Proceedings of the National Academy of Sciences* **89**(22): 10915–10919.
URL: *https://doi.org/10.1073/pnas.89.22.10915*

Herman, J. L., Szabó, A., Miklós, I. and Hein, J. (2015). Approximate statistical alignment by iterative sampling of substitution matrices, *arXiv preprint arXiv:1501.04986* .

Hogeweg, P. and Hesper, B. (1984). The alignment of sets of sequences and the construction of phyletic trees: an integrated method, *Journal of Molecular Evolution* **20**(2): 175–186.

Holm, L. and Sander, C. (1993). Protein structure comparison by alignment of distance matrices, *Journal of Molecular Biology* **233**(1): 123–138.

Holmes, I. (1998). *Studies in probabilistic sequence alignment and evolution*, PhD thesis, University of Cambridge.
URL: *http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.56.2236&rep=rep1&type=pdf*

Holmes, I. H. (2017). Solving the master equation for Indels, *BMC Bioinformatics* **18**(1): 1–10.

Holmes, I. and Rubin, G. (2002). An expectation maximization algorithm for training hidden substitution models, *Journal of Molecular Biology* **317**(5): 753–764.

Hough, B. (2003). Lecture 28: The Spectral Gap (Statistics 205b: Probability Theory).
URL: *https://www.stat.berkeley.edu/~pitman/s205s03/lecture28.pdf*

Hourai, Y., Akutsu, T. and Akiyama, Y. (2004). Optimizing substitution matrices by separating score distributions, *Bioinformatics* **20**(6): 863–873.

Huang, X. (2008). Sequence alignment with an appropriate substitution matrix, *Journal of Computational Biology* **15**(2): 129–138.

Huelsenbeck, J. P., Joyce, P., Lakner, C. and Ronquist, F. (2008). Bayesian analysis of amino acid substitution models, *Philosophical Transactions of the Royal Society B: Biological Sciences* **363**(1512): 3941–3953.

Huffman, D. A. (1952). A method for the construction of minimum-redundancy codes, *Proceedings of the IRE* **40**(9): 1098–1101.

IUPAC-IUB Committee on Biochemistry Nomenclature (1968). A one-letter notation for amino acid sequences. Tentative rules, *Biochemistry* **7**(8): 2703–2705.

Johnson, M. S. and Overington, J. P. (1993). A structural basis for sequence comparisons: an evaluation of scoring methodologies, *Journal of Molecular Biology* **233**(4): 716–738.

Jones, D. T., Taylor, W. R. and Thornton, J. M. (1992a). The rapid generation of mutation data matrices from protein sequences, *Bioinformatics* **8**(3): 275–282.

Jones, D. T., Taylor, W. and Thornton, J. M. (1992b). A new approach to protein fold recognition, *Nature* **358**(6381): 86.

Kachroo, A. H., Laurent, J. M., Yellman, C. M., Meyer, A. G., Wilke, C. O. and Marcotte, E. M. (2015). Systematic humanization of yeast genes reveals conserved functions and genetic modularity, *Science* **348**(6237): 921–925.

Kann, M., Qian, B. and Goldstein, R. A. (2000). Optimization of a new score function for the detection of remote homologs, *Proteins: Structure, Function, and Bioinformatics* **41**(4): 498–503.

Karlin, S. and Altschul, S. F. (1990). Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes, *Proceedings of the National Academy of Sciences* **87**(6): 2264–2268.

Kasarapu, P. (2015). Modelling of directional data using Kent distributions, *arXiv* .
URL: *http://arxiv.org/abs/1506.08105*

Katoh, K., Misawa, K., Kuma, K.-i. and Miyata, T. (2002). MAFFT: a novel method for rapid multiple sequence alignment based on fast fourier transform, *Nucleic Acids Research* **30**(14): 3059–3066.

Katoh, K. and Standley, D. M. (2013). MAFFT multiple sequence alignment software version 7: improvements in performance and usability, *Molecular Biology and Evolution* **30**(4): 772–780.

Keane, T. M., Creevey, C. J., Pentony, M. M., Naughton, T. J. and Mclnerney, J. O. (2006). Assessment of methods for amino acid matrix selection and their use on empirical data shows that ad hoc assumptions for choice of matrix are not justified, *BMC Evolutionary Biology* **6**(1): 29.

Keul, F., Hess, M., Goesele, M. and Hamacher, K. (2017). PFASUM: a substitution matrix from Pfam structural alignments, *BMC Bioinformatics* **18**(1): 293.

Kimura, M. (1983). *The Neutral Theory of Molecular Evolution*, Cambridge University Press.

Kirkpatrick, S., Gelatt, C. D. and Vecchi, M. P. (1983). Optimization by simulated annealing, *Science* **220**(4598): 671–680.

Kishino, H., Miyata, T. and Hasegawa, M. (1990). Maximum likelihood inference of protein phylogeny and the origin of chloroplasts, *Journal of Molecular Evolution* **31**(2): 151–160.

Koehl, P. (2006). Protein structure classification, *Reviews in Computational Chemistry* **22**: 1.

Kolaskar, A. and Kulkarni-Kale, U. (1992). Sequence alignment approach to pick up conformationally similar protein fragments, *Journal of Molecular Biology* **223**(4): 1053–1061.

Konagurthu, A. S., Kasarapu, P., Allison, L., Collier, J. H. and Lesk, A. M. (2015). On sufficient statistics of least-squares superposition of vector sets, *Journal of Computational Biology* **22**(6): 487–497.
URL: *https://doi.org/10.1007/978-3-319-05269-4_11*

Konagurthu, A. S., Lesk, A. M. and Allison, L. (2012). Minimum message length inference of secondary structure from protein coordinate data, *Bioinformatics* **28**(12): i97–i105.
URL: *https://doi.org/10.1093/bioinformatics/bts223*

Konagurthu, A. S., Whisstock, J. C., Stuckey, P. J. and Lesk, A. M. (2006). MUSTANG: a multiple structural alignment algorithm, *Proteins: Structure, Function, and Bioinformatics* **64**(3): 559–574.
URL: *https://doi.org/10.1002/prot.20921*

Kosiol, C. and Goldman, N. (2005). Different versions of the Dayhoff rate matrix, *Molecular Biology and Evolution* **22**(2): 193–199.

Kosiol, C. and Goldman, N. (2011). Markovian and non-Markovian protein sequence evolution: aggregated Markov process models, *Journal of Molecular Biology* **411**(4): 910–923.

Krause, A., Stoye, J. and Vingron, M. (2000). The SYSTERS protein sequence cluster set, *Nucleic Acids Research* **28**(1): 270–272.

Kullback, S. and Leibler, R. A. (1951). On information and sufficiency, *The Annals of Mathematical Statistics* **22**(1): 79–86.

Larkin, M. e. a. (2007). Clustal W and Clustal X version 2.0, *Bioinformatics* **23**(21): 2947–2948.

Lassmann, T. and Sonnhammer, E. L. (2005). Kalign–an accurate and fast multiple sequence alignment algorithm, *BMC Bioinformatics* **6**(1): 1–9.

Lathrop, R. H., Rogers, R. G., Smith, T. F. and White, J. V. (1998). A Bayes-optimal sequence-structure theory that unifies protein sequence-structure recognition and alignment, *Bulletin of Mathematical Biology* **60**(6): 1039–1071.

Le, S. Q. and Gascuel, O. (2008). An improved general amino acid replacement matrix, *Molecular Biology and Evolution* **25**(7): 1307–1320.

Lefranc, M.-P., Giudicelli, V., Duroux, P., Jabado-Michaloud, J., Folch, G., Aouinti, S., Carillon, E., Duvergey, H., Houles, A., Paysan-Lafosse, T. et al. (2015). IMGT®, the international ImMunoGeneTics information system 25 years on, *Nucleic Acids Research* **43**(D1): D413–D422.

Lesk, A. M. (2001). *Introduction to Protein Architecture: The Structural Biology of Proteins*, Oxford University Press.

Lesk, A. M. (2010). *Introduction to Genomics*, Oxford University Press.

Lesk, A. M. (2016). *Introduction to Protein Science: Architecture, Function, and Genomics*, Oxford University Press.

Lesk, A. M., Subramanian, R., Allison, L., Abramson, D., Stuckey, P. J., de la Banda, M. G. and Konagurthu, A. S. (2018). Universal architectural concepts underlying protein folding patterns, *bioRxiv* p. 480194.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals, *Soviet Physics Doklady*, Vol. 10, pp. 707–710.

Levin, J. M., Robson, B. and Garnier, J. (1986). An algorithm for secondary structure determination in proteins based on sequence similarity, *FEBS Letters* **205**(2): 303–308.

Levinthal, C. (1969). How to fold graciously, *Mössbaun Spectroscopy in Biological Systems Proceedings* **67**(41): 22–24.

Levitt, M. and Gerstein, M. (1998). A unified statistical framework for sequence comparison and structure comparison, *Proceedings of the National Academy of Sciences* **95**(11): 5913–5920.

Levy Karin, E., Ashkenazy, H., Hein, J. and Pupko, T. (2018). A simulation-based approach to statistical alignment, *Systematic Biology* .

Li, W., Cerise, J. E., Yang, Y. and Han, H. (2017). Application of t-SNE to human genetic data, *Journal of Bioinformatics and Computational Biology* **15**(04): 1750017.

Linderstrøm-Lang, K. U. (1952). *Lane Medical Lectures: Proteins and Enzymes*, Vol. 6, Stanford University Press.

Lipman, D. J. and Pearson, W. R. (1985). Rapid and sensitive protein similarity searches, *Science* **227**(4693): 1435–1441.

Löytynoja, A. and Goldman, N. (2008). Phylogeny-aware gap placement prevents errors in sequence alignment and evolutionary analysis, *Science* **320**(5883): 1632–1635.

Lüthy, R., McLachlan, A. D. and Eisenberg, D. (1991). Secondary structure-based profiles: Use of structure-conserving scoring tables in searching protein sequence databases for structural similarities, *Proteins: Structure, Function, and Bioinformatics* **10**(3): 229–239.

Marsh, J. A. and Teichmann, S. A. (2010). How do proteins gain new domains?, *Genome Biology* **11**(7): 126.

McLachlan, A. D. (1971). Tests for comparing related amino-acid sequences. Cytochrome c and cytochrome c551, *Journal of Molecular Biology* **61**(2): 409–424.

Menke, M., Berger, B. and Cowen, L. (2008). Matt: local flexibility aids protein multiple structure alignment, *PLOS Computational Biology* **4**(1): e10.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. and Teller, E. (1953). Equation of state calculations by fast computing machines, *The Journal of Chemical Physics* **21**(6): 1087–1092.

Mi, H., Lazareva-Ulitsky, B., Loo, R., Kejariwal, A., Vandergriff, J., Rabkin, S., Guo, N., Muruganujan, A., Doremieux, O., Campbell, M. J. et al. (2005). The PANTHER database of protein families, subfamilies, functions and pathways, *Nucleic Acids Research* **33**(suppl_1): D284–D288.

Miko, I. and LeJeune, L. (2009). Essentials of Genetics, *Cambridge: NPG Education* .

Miller, W. and Myers, E. W. (1988). Sequence comparison with concave weighting functions, *Bulletin of Mathematical Biology* **50**(2): 97–120.

Miyata, T., Miyazawa, S. and Yasunaga, T. (1979). Two types of amino acid substitutions in protein evolution, *Journal of Molecular Evolution* **12**(3): 219–236.

Miyazawa, S. and Jernigan, R. L. (1993). A new substitution matrix for protein sequence searches based on contact frequencies in protein structures, *Protein Engineering, Design and Selection* **6**(3): 267–278.

Mizuguchi, K., Deane, C., Blundell, T. and Overington, J. (1998). HOMSTRAD: a database of protein structure alignments for homologous families, *Protein Science* **7**(11): 2469–2471.

Mount, D. W. (2007). Steps used by the BLAST algorithm, *Cold Spring Harbor Protocols* **2007**(7).
URL: *http://cshprotocols.cshlp.org/content/2007/7/pdb.ip41.abstract*

Müller, T., Rahmann, S. and Rehmsmeier, M. (2001). Non-symmetric score matrices and the detection of homologous transmembrane proteins, *Bioinformatics* **17**(suppl_1): S182–S189.

Müller, T., Spang, R. and Vingron, M. (2002). Estimating amino acid substitution models: a comparison of Dayhoff's estimator, the resolvent approach and a maximum likelihood method, *Molecular Biology and Evolution* **19**(1): 8–13.

Müller, T. and Vingron, M. (2000). Modeling amino acid replacement, *Journal of Computational Biology* **7**(6): 761–776.

Murzin, A. G., Brenner, S. E., Hubbard, T. and Chothia, C. (1995). SCOP: a structural classification of proteins database for the investigation of sequences and structures, *Journal of Molecular Biology* **247**(4): 536–540.

Ndhlovu, A., Hazelhurst, S. and Durand, P. M. (2015). Robust sequence alignment using evolutionary rates coupled with an amino acid substitution matrix, *BMC Bioinformatics* **16**(1): 255.

Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins, *Journal of Molecular Biology* **48**(3): 443–453.

Nei, M. and Kumar, S. (2000). *Molecular Evolution and Phylogenetics*, Oxford University Press.

Ng, P. C., Henikoff, J. G. and Henikoff, S. (2000). PHAT: a transmembrane-specific substitution matrix, *Bioinformatics* **16**(9): 760–766.

Nickle, D. C., Heath, L., Jensen, M. A., Gilbert, P. B., Mullins, J. I. and Pond, S. L. K. (2007). Hiv-specific probabilistic models of protein evolution, *PLOS ONE* **2**(6): e503.

Niefind, K. and Schomburg, D. (1991). Amino acid similarity coefficients for protein modeling and sequence alignment derived from main-chain folding angles, *Journal of Molecular Biology* **219**(3): 481–497.

Norris, J. R. (1998). *Markov chains*, second edn, Cambridge University Press.

Notredame, C., Higgins, D. G. and Heringa, J. (2000). T-Coffee: a novel method for fast and accurate multiple sequence alignment1, *Journal of Molecular Biology* **302**(1): 205–217.

Oliver, J. and Hand, D. (1994). Introduction to minimum encoding inference, *Technical Report 4-94*, Statistics Dept., Open University.

Orengo, C. A., Michie, A. D., Jones, S., Jones, D. T., Swindells, M. B. and Thornton, J. M. (1997). CATH–a hierarchic classification of protein domain structures, *Structure* **5**(8): 1093–1109.

Ota, T. and Nei, M. (1994). Estimation of the number of amino acid substitutions per site when the substitution rate varies among sites, *Journal of Molecular Evolution* **38**(6): 642–643.

Pang, A., Smith, A. D., Nuin, P. A. and Tillier, E. R. (2005). SIMPROT: using an empirically determined indel distribution in simulations of protein evolution, *BMC Bioinformatics* **6**(1): 1–7.

Pascarella, S. and Argos, P. (1992). Analysis of insertions/deletions in protein structures, *Journal of Molecular Biology* **224**(2): 461–471.

Pauling, L. and Corey, R. B. (1951). The pleated sheet, a new layer configuration of polypeptide chains, *Proceedings of the National Academy of Sciences* **37**(5): 251.

Pauling, L., Corey, R. B. and Branson, H. R. (1951). The structure of proteins: Two hydrogen-bonded helical configurations of the polypeptide chain, *Proceedings of the National Academy of Sciences* **37**(4): 205–211.

Pearson, W. R. and Lipman, D. J. (1988). Improved tools for biological sequence comparison, *Proceedings of the National Academy of Sciences* **85**(8): 2444–2448.

Perron, O. (1907). Zur theorie der matrices, *Mathematische Annalen* **64**(2): 248–263.

Pietrokovski, S., Henikoff, J. G. and Henikoff, S. (1996). The Blocks database—a system for protein classification, *Nucleic Acids Research* **24**(1): 197–200.

Platzer, A. (2013). Visualization of SNPs with t-SNE, *PLOS ONE* **8**(2): e56883.

Pollock, D. D., Pollard, S. T., Shortt, J. A. and Goldstein, R. A. (2017). Mechanistic models of protein evolution, *Evolutionary Biology: Self/Nonself Evolution, Species and Complex Traits Evolution, Methods and Concepts*, Springer, pp. 277–296.

Powell, D. R., Allison, L. and Dix, T. I. (2004). Modelling-alignment for non-random sequences., *Australian Conference on Artificial Intelligence*, Springer, pp. 203–214.
URL: *https://doi.org/10.1007/978-3-540-30549-1_19*

Qian, B. and Goldstein, R. A. (2001). Distribution of indel lengths, *Proteins: Structure, Function, and Bioinformatics* **45**(1): 102–104.

Qian, B. and Goldstein, R. A. (2002). Optimization of a new score function for the generation of accurate alignments, *Proteins: Structure, Function, and Bioinformatics* **48**(4): 605–610.

Qiu, J. and Elber, R. (2006). SSALN: an alignment algorithm using structure-dependent substitution matrices and gap penalties learned from structurally aligned protein pairs, *Proteins: Structure, Function, and Bioinformatics* **62**(4): 881–891.

Rao, J. M. (1987). New scoring matrix for amino acid residue exchanges based on residue characteristic physical parameters, *International Journal of Peptide and Protein Research* **29**(2): 276–281.

Redelings, B. D. and Suchard, M. A. (2005). Joint Bayesian estimation of alignment and phylogeny, *Systematic Biology* **54**(3): 401–418.

Rice, D. W. and Eisenberg, D. (1997). A 3D-1D substitution matrix for protein fold recognition that includes predicted secondary structure of the sequence, *Journal of Molecular Biology* **267**(4): 1026–1038.

Richardson, J. S. (1981). Protein anatomy, *Advances in Protein Chemistry* **34**(167): 339.

Risler, J., Delorme, M., Delacroix, H. and Henaut, A. (1988). Amino acid substitutions in structurally related proteins a pattern recognition approach: Determination of a new and efficient scoring matrix, *Journal of Molecular Biology* **204**(4): 1019–1029.

Rivas, E. and Eddy, S. R. (2015). Parameterizing sequence alignment with an explicit evolutionary model, *BMC Bioinformatics* **16**(1): 406.

Rosenberg, M. S. (2009). *Sequence Alignment: Methods, Models, Concepts, and Strategies*, University of California Press.

Rost, B. (1997). Protein structures sustain evolutionary drift, *Folding and Design* **2**: S19–S24.

Rost, B. (1999). Twilight zone of protein sequence alignments, *Protein Engineering* **12**(2): 85–94.

Rozewicki, J., Li, S., Amada, K. M., Standley, D. M. and Katoh, K. (2019). MAFFT-DASH: integrated protein sequence and structural alignment, *Nucleic Acids Research* **47**(W1): W5–W10.

Russell, R. B. and Barton, G. J. (1992). Multiple protein sequence alignment from tertiary structure comparison: assignment of global and residue confidence levels, *Proteins: Structure, Function, and Bioinformatics* **14**(2): 309–323.

Russell, R. B., Saqi, M. A., Sayle, R. A., Bates, P. A. and Sternberg, M. J. (1997). Recognition of analogous and homologous protein folds: analysis of sequence and structure conservation, *Journal of Molecular Biology* **269**(3): 423–439.

Saigo, H., Vert, J.-P. and Akutsu, T. (2006). Optimizing amino acid substitution matrices with a local alignment kernel, *BMC Bioinformatics* **7**(1): 246.

Sali, A. and Blundell, T. L. (1990). Definition of general topological equivalence in protein structures: A procedure involving comparison of properties and relationships through simulated annealing and dynamic programming, *Journal of Molecular Biology* **212**(2): 403–428.

Sanger, F. and Tuppy, H. (1951). The amino-acid sequence in the phenylalanyl chain of insulin. 1. the identification of lower peptides from partial hydrolysates, *Biochemical Journal* **49**(4): 463–481.

Saraswathy, N. and Ramalingam, P. (2011). *Concepts and Techniques in Genomics and Proteomics*, Elsevier.

Sarich, V. M. and Wilson, A. C. (1967). Immunological time scale for hominid evolution, *Science* **158**(3805): 1200–1203.

Schrödinger Inc. (2015). *The PyMOL Molecular Graphics System, Version 2.0.7.*

Shannon, C. E. (1948). A mathematical theory of communication, *Bell System Technical Journal* **27**: 379–423.

Shindyalov, I. N. and Bourne, P. E. (1998). Protein structure alignment by incremental combinatorial extension (CE) of the optimal path., *Protein Engineering* **11**(9): 739–747.

Slater, A. W., Castellanos, J. I., Sippl, M. J. and Melo, F. (2012). Towards the development of standardized methods for comparison, ranking and evaluation of structure alignments, *Bioinformatics* **29**(1): 47–53.

Smith, T. F., Lo Conte, L., Bienkowska, J., Gaitatzes, C., Rogers, R. G. and Lathrop, R. (1997). Current limitations to protein threading approaches, *Journal of Computational Biology* **4**(3): 217–225.

Smith, T. F., Waterman, M. S. et al. (1981). Identification of common molecular subsequences, *Journal of Molecular Biology* **147**(1): 195–197.

Soskine, M. and Tawfik, D. S. (2010). Mutational effects and the evolution of new protein functions, *Nature Reviews Genetics* **11**(8): 572–582.

Stewart, W. J. (1994). *Introduction to the numerical solution of Markov chains*, Princeton University Press.

Sumanaweera, D., Allison, L. and Konagurthu, A. (2018). The bits between proteins, *2018 Data Compression Conference*, IEEE, pp. 177–186.
URL: *https://doi.org/10.1109/DCC.2018.00026*

Sumanaweera, D., Allison, L. and Konagurthu, A. S. (2019). Statistical compression of protein sequences and inference of marginal probability landscapes over competing alignments using finite state models and Dirichlet priors, *Bioinformatics* **35**(14): i360–i369.
URL: *https://doi.org/10.1093/bioinformatics/btz368*

Sumanaweera, D., Allison, L. and Konagurthu, A. S. (2020). Bridging the gaps in statistical models of protein alignment, *arXiv preprint arXiv:2010.00855* .
URL: *https://arxiv.org/abs/2010.00855*

Sutcliffe, M. J., Haneef, I., Carney, D. and Blundell, T. (1987). Knowledge based modelling of homologous proteins, Part I: Three-dimensional frameworks derived from the simultaneous superposition of multiple structures, *Protein Engineering, Design and Selection* **1**(5): 377–384.

Swanson, R. (1984). A unifying concept for the amino acid code, *Bulletin of Mathematical Biology* **46**(2): 187–203.

Szklarczyk, R., Wanschers, B. F., Cuypers, T. D., Esseling, J. J., Riemersma, M., van den Brand, M. A., Gloerich, J., Lasonder, E., van den Heuvel, L. P., Nijtmans, L. G. et al. (2012). Iterative orthology prediction uncovers new mitochondrial proteins and identifies C12orf62 as the human ortholog of COX14, a protein involved in the assembly of cytochrome c oxidase, *Genome Biology* **13**(2): R12.

Tan, Y. H., Huang, H. and Kihara, D. (2006). Statistical potential-based amino acid similarity matrices for aligning distantly related protein sequences, *Proteins: Structure, Function, and Bioinformatics* **64**(3): 587–600.

Tatusova, T. A. and Madden, T. L. (1999). BLAST 2 Sequences, a new tool for comparing protein and nucleotide sequences, *FEMS Microbiology Letters* **174**(2): 247–250.

The Mathworks, Inc. (2017). *MATLAB Version 9.2.0.556344 (R2017a)*, Natick, Massachusetts.

Tomii, K. and Kanehisa, M. (1996). Analysis of amino acid indices and mutation matrices for sequence comparison and structure prediction of proteins, *Protein Engineering, Design and Selection* **9**(1): 27–36.

Truszkowski, J. and Brown, D. G. (2011). More accurate recombination prediction in HIV-1 using a robust decoding algorithm for HMMs, *BMC Bioinformatics* **12**(1): 1–11.

UniProt Consortium and others (2017). UniProt: the universal protein knowledgebase, *Nucleic Acids Research* **45**(D1): D158–D169.

van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-SNE, *Journal of Machine Learning Research* **9**(Nov): 2579–2605.

Van Walle, I., Lasters, I. and Wyns, L. (2005). SABmark—a benchmark for sequence alignment that covers the entire known fold space, *Bioinformatics* **21**(7): 1267–1268.

Veerassamy, S., Smith, A. and Tillier, E. R. (2003). A transition probability model for amino acid substitutions from blocks, *Journal of Computational Biology* **10**(6): 997–1010.

Venclovas, Č. (2011). Methods for sequence–structure alignment, *Homology Modeling*, Springer, pp. 55–82.

Vingron, M. and Waterman, M. S. (1994). Sequence alignment and penalty choice: Review of concepts, case studies and implications, *Journal of Molecular Biology* **235**(1): 1–12.

Vogel, C., Bashton, M., Kerrison, N. D., Chothia, C. and Teichmann, S. A. (2004). Structure, function and evolution of multidomain proteins, *Current Opinion in Structural Biology* **14**(2): 208–216.

Vogt, G., Etzold, T. and Argos, P. (1995). An assessment of amino acid exchange matrices in aligning protein sequences: the twilight zone revisited, *Journal of Molecular Biology* **249**(4): 816–831.

Wallace, C. S. (2005). *Statistical and Inductive Inference by Minimum Message Length*, Information Science and Statistics, Springer.
URL: https://doi.org/10.1007/0-387-27656-4

Wallace, C. S. and Boulton, D. M. (1968). An information measure for classification, *Computer Journal* **11**(2): 185–194.
URL: https://doi.org/10.1093/comjnl/11.2.185

Wallace, C. S. and Dowe, D. L. (1993). MML estimation of the von Mises concentration parameter, *Technical report*, Monash University.

Wallace, C. S. and Freeman, P. R. (1987). Estimation and inference by compact coding, *Journal of the Royal Statistical Society: Series B (Methodological)* **49**(3): 240–265.
URL: http://www.jstor.org/stable/2985992

Wallace, C. S. and Patrick, J. D. (1993). Coding decision trees, *Machine Learning* **11**(1): 7–22.
URL: http://dx.doi.org/10.1023/A:1022646101185

Whelan, S. and Goldman, N. (2001). A general empirical model of protein evolution derived from multiple protein families using a maximum-likelihood approach, *Molecular Biology and Evolution* **18**(5): 691–699.

Yamada, K. and Tomii, K. (2013). Revisiting amino acid substitution matrices for identifying distantly related proteins, *Bioinformatics* **30**(3): 317–325.

Yang, Z., Nielsen, R. and Hasegawa, M. (1998). Models of amino acid substitution and applications to mitochondrial protein evolution., *Molecular Biology and Evolution* **15**(12): 1600–1611.

Ye, X., Yu, Y.-K. and Altschul, S. F. (2011). On the inference of Dirichlet mixture priors for protein sequence comparison, *Journal of Computational Biology* **18**(8): 941–954.

Yee, C. N. and Allison, L. (1993). Reconstruction of strings past, *Bioinformatics* **9**(1): 1–7.
URL: https://doi.org/10.1093/bioinformatics/9.1.1

Yona, G. and Levitt, M. (2000). A unified sequence-structure classification of protein sequences: combining sequence and structure in a map of the protein space, *Proceedings of the fourth annual International Conference on Computational Molecular Biology*, pp. 308–317.

Yu, Y. and Altschul, S. F. (2004). The construction of amino acid substitution matrices for the comparison of proteins with non-standard compositions, *Bioinformatics* **21**(7): 902–911.

Yu, Y., Wootton, J. C. and Altschul, S. F. (2003). The compositional adjustment of amino acid substitution matrices, *Proceedings of the National Academy of Sciences* **100**(26): 15688–15693.

Zahn, L. M. (2014). How proteins can evolve new functions, *Science* **345**: 634–634.

Zuckerkandl, E. and Pauling, L. (1965). Evolutionary divergence and convergence in proteins, *Evolving Genes and Proteins* **97**: 97–166.