ROBUST SEQUENCE ALIGNMENT USING EVOLUTIONARY RATES COUPLED WITH AN AMINO ACID SUBSTITUTION MATRIX

Andrew Ndhlovu

Supervisors:

Pierre M. Durand
and
Scott Hazelhurst

A dissertation submitted to the Faculty of Health Sciences, University of the Witwatersrand, Johannesburg, in fulfilment of the requirements of the degree of
Master of Science (Medicine).

Johannesburg, 2014

Declaration

I, Andrew Ndhlovu declare that this dissertation is my own work. It is being submitted for the degree of Master of Science (Medicine) at the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination at this or any other University.

.....

..... day of ....., 2015

Dedication

To Ephinah and Stephen Ndhlovu, my father and mother. To Rostar, thank you for inspiring and believing in me.

Publications and presentations arising from this study

Manuscripts in review

- **Ndhlovu, A.**, Durand, P. M. and Hazelhurst, S. *Database*. EvoDB: A database of evolutionary rate profiles, associated protein domains and phylogenetic trees for PFAM-A. manuscript submitted.

- **Ndhlovu, A.**, Hazelhurst, S and Durand, P. M. *BMC Bioinformatics*. Robust sequence alignment using evolutionary rates coupled with an amino acid substitution matrix. manuscript submitted.

Presentations

- **Ndhlovu, A.**, Hazelhurst, S and Durand, P. M. International Society for Computational Biology and African Society of Computational Biology (ISCB-ASCB) conference. Robust sequence alignment through an evolutionary rate coupled with substitution matrix approach. 15 March 2015. Abstract submitted for oral presentation.

- **Ndhlovu, A.**, Hazelhurst, S and Durand, P. M. Evolutionary rates infer protein domain function through sequence alignment. Poster Presentation at the Molecular Biosciences Research Thrust, University of the Witwatersrand. 5 December 2014.

- **Ndhlovu, A.**, Hazelhurst, S and Durand, P. M. Evolutionary rates infer protein domain function through sequence alignment. Poster Presentation at the University of the Witwatersrand, 6th Cross Faculty Graduate Symposium. 28 to 29 October 2014.

- **Ndhlovu, A.**, Hazelhurst, S and Durand, P. M. Evolutionary rates infer protein domain function through sequence alignment. Poster Presentation at the Second Joint Congress of the South African Society for Bioinformatics (SASBi) and the South African Genetics Society (SAGS), 23-26 September 2014.

- **Ndhlovu, A.**, Hazelhurst, S and Durand, P. M. Evolutionary rates infer protein domain function through sequence alignment. Oral Presentation at the University of the Witwatersrand, Health Sciences Biennial Research Day and Postgraduate Expo 2014, 17 September 2014.

- **Ndhlovu, A.**, Hazelhurst, S and Durand, P. M. Evolutionary rates infer protein domain function through sequence alignment. Oral presentation at the University of the Witwatersrand, Molecular Medicine and Haematology Seminar Series. 6 August 2014

- **Ndhlovu, A.**, Hazelhurst, S and Durand, P. M. Improving fire: functional inference using rates of evolution reinforced by the BLOSUM substitution matrix. Molecular Biosciences Research Thrust. 5 December 2012.

Abstract

Selective pressures at the DNA level shape genes into profiles consisting of patterns of rapidly evolving sites and sites withstanding change. These profiles remain detectable even when protein sequences become extensively diverged. It has been hypothesised that these patterns can be used as gene identifiers. A common task in molecular biology is to infer functional, structural or evolutionary relationships by querying a database using an algorithm. However, problems arise when sequence similarity is low. This study presents an algorithm that uses evolutionary rates at codon sites inferred from the d_N/d_S (ω) parameter as an alignment metric for detecting distantly related proteins. The current version of the algorithm, FIRE (***F***unctional ***I***nference using ***R***ates of **E**volution), uses evolutionary rates as a proxy for selective pressure to address the challenge of low sequence similarity. The problem is that the algorithm produces numerous false positives when highly conserved datasets are aligned. To increase the sensitivity of the algorithm, the evolutionary rate based approach was reimplemented and coupled with a conventional BLOSUM substitution matrix to produce a new implementation called BLOSUM-FIRE. The two approaches are combined in a dynamic scoring function, which uses the selective pressure to score aligned residues. Analysis of quality of alignments produced, revealed that the new implementation of the FIRE algorithm performs as well as conventional algorithms. In addition, the Evolutionary rate Database (EvoDB), which is a compilation of evolutionary rate profiles of all the members of the PFAM-A protein domain database has been developed. The EvoDB database can be queried using FIRE to infer protein domain functions. The utility of this algorithm and database was tested by inferring the domain functions of the Hepatitis B X protein. Results show that the BLOSUM-FIRE algorithm was able to accurately identify the domain function of HBx as a trans-activation protein using EvoDB. The algorithm also found statistically significant ($p = 7.6 \times 10^{-5}$) similarity between the HBx the rubella virus endopeptidase protein. The biological relevance of these results was not validated and requires further interrogation; however, these proteins share vital roles in viral replication. This study demonstrates the utility of an evolutionary rate based approach and demonstrates that such an approach is robust when coupled with an amino acid substitution matrix yielding results comparable to conventional algorithms. EvoDB is a catalogue of the evolutionary rate profiles and provides the corresponding phylogenetic trees, PFAM-A alignments and annotated accession identifier data. The BLOSUM-FIRE software and user manual including the EvoDB flat file database and release notes have been made freely available at www.bioinf.wits.ac.za/software/fire. The BLOSUM-FIRE algorithm and EvoDB database present a tier of information untapped by current databases and tools.

Acknowledgments

I thank Dr Pierre M. Durand and Professor Scott Hazelhurst. They were very supportive throughout the project, Dr Pierre M. Durand for his expert advice on evolution and Professor Scott Hazelhurst for his expert guidance on implementing the algorithm and compiling the EvoDB database. I thank Liam J. Thompson who provided the curated HBx nucleotides sequences that were used in this study. The financial assistance of the National Research Foundation (NRF) towards this research is hereby acknowledged. Opinions expressed and conclusions arrived at, are those of the author and are not necessarily to be attributed to the NRF.

Johannesburg, October 2014

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The initial step for the inference of phylogenetic relationships or functions of proteins is using a sequence alignment analysis with tools such as the ***B***asic ***L***ocal ***A***lignment ***S***earch ***T***ool (BLAST) (Altschul et al., 1990) to search a biological database for example GenBank (Benson et al., 2010). Once a statistically significant match has been made with a known protein, the putative functions or evolutionary relationships (Kuzniar et al., 2008) can then be inferred for the query sequence. However, challenges arise when sequence similarity is low and inferences cannot be made (Rost, 1999). This region of low sequence similarity in the region of 20% and 30%, in which sequence relatedness cannot be inferred has been designated the "twilight zone" in sequence alignment (Doolittle, 1981; Rost, 1999). On the other hand, structural information has been used to infer sequence relatedness with similarities as low as 30%, for example, globin proteins (Chothia and Lesk, 1986).

Evolutionary pressures resulting from natural selection at the DNA level have been found to mould genes into patterns of sites that are highly conserved (resistant to change) and those sites that are poorly conserved (evolving rapidly). The level of conservation can be inferred from the ratio of the non-synonymous substitution rate (d_N or Ka) to the synonymous substitution rate (d_S or Ks) loosely referred to as the evolutionary rate, represented by the parameter ω or d_N/d_S (Kimura, 1984). These

patterns of evolutionary rates or "Evolutionary fingerprints" (Pond et al., 2010) can be used as a similarity metric in a sequence alignment algorithm (Durand et al., 2010).

1.1 Problem statement

A novel alignment algorithm, ***F***unctional ***I***nference using ***R***ates of ***E***volution (FIRE), was developed to address the low similarity challenge (Durand et al., 2010). FIRE uses the evolutionary rate ($\omega = d_N/d_S$) at codon sites, rather than individual amino acid residue identities to align sequences thus circumventing the problem of low sequence similarity. However, the preliminary analysis revealed numerous false positives, particularly when two unrelated highly conserved domains were aligned. Additionally, a database for searching and inferring functions of query proteins is required.

1.2 Research Objectives

This research aims to conceptualise an approach to implement an improved and refined version of the FIRE algorithm and to develop a database for inferring protein domain functions. Therefore, the objectives of this research are:

1. To evaluate the performance in term of quality of alignments of the FIRE algorithm using alignments from tools like CLUSTAL Omega, MAFFT, MUSCLE and T-COFFEE. An approach with a focus on the algorithm scoring function and gap penalties, using residue matches as a performance measure will be used.
2. To conceptualise an approach to improve and refine the algorithm.
3. To iteratively modify and evaluate the FIRE algorithm until the performance is comparable to that of conventional algorithms.
4. To compile an evolutionary rates database (EvoDB) using Maximum Likelihood Estimates (MLEs) of d_N/d_S at codon sites of the PFAM-A Protein domain database.
5. To identify and retrieve HBx coding sequences and use the CODEML program

found on the PAML suite of software to determine the MLEs of ω at codon sites for the Hepatitis B Virus X (HBx) protein.

6. To use the FIRE algorithm to search EvoDB for statistically significant similarities with the HBx protein and ultimately infer its protein domain functions.

1.3 Road Map

This study describes an approach to address the biological challenges of aligning proteins to infer domain functions using algorithms and related tools. An evolutionary rate based approach is provided in this research to address this challenge. Therefore, this research describes:

i. Conceptualisation and formulation of a new algorithm.
ii. Implementation and development of a new algorithm.
iii. Compilation of an evolutionary rates database.
iv. A test case for the new algorithm and the database.

Chapter 2 provides a brief background on the challenges of sequence alignment, algorithms that have been proposed and the various approaches considered in their design. The chapter evaluates and describes the different strategies used in sequence alignment algorithms. The chapter does not attempt to give an exhaustive study of sequence alignment; however, it provides a review of current approaches and strategies including theoretical frameworks on which they are based. The chapter reviews the important parameters for consideration when implementing an algorithm and discusses some of the novel approaches that have been proposed. The review also discusses the challenges presented by the low sequence similarity problem.

Chapter 3 begins by formulating the design of the algorithm. This is followed by a description of the approach used to evaluate the algorithm and how its limitations

were identified. The chapter gives an account of how the FIRE algorithm was coupled with the BLOSUM62 matrix to produce a new sequence alignment algorithm called BLOSUM-FIRE. Experiments and data sets used to evaluate the implementation of the new algorithm are described and presented. The empirical determination of optimal gap penalties based on a variable gap penalty scheme is also provided. This chapter also presents the compilation of the Evolutionary Rate Database (EvoDB) for the BLOSUM-FIRE algorithm. Finally the chapter describes how the EvoDB can be used to infer protein domain function for proteins such as the Hepatitis B Virus X protein.

Chapter 4 reports on the results of the FIRE algorithm evaluation experiments. The results of the conceptualisation of the BLOSUM-FIRE algorithm are presented. The performance of BLOSUM-FIRE algorithm is assessed by comparison with widely used sequence alignment algorithms. The chapter also presents results on the implementation and development of the EvoDB database.

Chapter 5 discusses the different approaches used in this research and evaluates their efficacy in solving the challenge of low sequence alignment in the twilight zone of sequence alignment and bioinformatics. In this chapter, some of the approaches used to implement the new algorithm are evaluated and how these can be improved is discussed.

In Chapter 6 conclusions about the BLOSUM-FIRE algorithm are drawn and the objectives of the study are addressed.

Chapter 2

Background

This chapter reviews related studies and background literature in addressing the challenges of sequence alignment, specifically the "twilight zone" in sequence alignment is discussed. The chapter reviews challenges of sequence alignment and the various algorithms that have been proposed to address them. Substitution matrices and gap penalties are reviewed; how these are implemented in sequence alignment algorithms is also discussed. The evolutionary rate (ω) and its utility in sequence alignment in relation to the conceptualization of the FIRE algorithm is discussed. Additionally, databases and how they form a vital tool in sequence analysis is also reviewed. Finally, as a test case the chapter provides a review of the Hepatitis B Virus X protein and the controversies surrounding its elusive functions. The chapter is concluded by discussing how tools like the one developed in this study may provide alternative approaches to inferring the functions of proteins like HBx.

2.1 Sequence alignment

The most common analysis in bioinformatics is performing a sequence alignment using an algorithm (Pearson and Lipman, 1988; Altschul et al., 1990; Durand et al., 2010; Di Tommaso et al., 2011) or inferring the function of a query sequence using a biological sequence database (Bateman, 2004; Benson et al., 2010). Results of such an analysis can then be used to infer evolutionary (Kuzniar et al., 2008), functional (Bateman,

2004) and structural (Murzin et al., 1995) relationships. Sequence alignment is an attempt to address the biological question of homology; sequence similarity is the metric and homology is the hypothesis. The distinction between the two: homology and similarity must not be confused; homology is inferred from sequence similarity, they are not equivalent.

Sequence alignment algorithms identify sites of similarity between the aligned letters of nucleotides or amino acid sequences. This study focuses on the alignment of amino acids which is considerably more challenging than that of nucleotide sequences. An amino acid sequence is a string over a twenty-letter alphabet such that:

$$AminoAlph = \{A - Y\} - \{B, J, O, U, X\},$$

where $AminoAlph$ is the amino acid alphabet. The objective of a sequence alignment analysis is to find regions of similarity between two sequences and identify the mathematically optimal alignment based on some objective scoring function. To model mutational events such as substitutions, insertions or deletions between sequences, gap characters ,"-" (hyphen) or "."(dot) are used.

The two commonly used approaches in pairwise comparison of sequences are: local and global alignments (Gotoh, 1982; Carrillo and Lipman, 1988). These approaches are usually implemented using a dynamic programming algorithm discussed in detail in Sections 2.1.1 and 2.1.2. Global alignments find similarities that span the whole length of the sequence and local alignments identify stretches or regions of similarity between sequences. The decision of which algorithm to use depends on some prior knowledge or assumption about the sequences and regions to be aligned.

Therefore, sequence alignment is a biological problem that has been solved using computationally based approaches. The frequent challenge is to test the hypothesis of homology between sequences in which sequence similarity is used as a metric. The

next two sections review and describe the local and global alignment approaches that underlie sequence alignment algorithms.

2.1.1 Global alignment algorithms

The dynamic programming (DP) algorithm is the optimisation engine of the sequence alignment algorithm responsible for assigning scores to aligned amino acids and subsequently the insertion of gaps to model mutational events such as insertions, deletions and substitutions. The Needleman-Wunsch (Needleman and Wunsch, 1970) algorithm is an implementation of a DP approach that was proposed to simplify the alignment problem by breaking the task into smaller tasks making it tractable. Global alignment approaches are implemented using the DP algorithm and have found use in detecting shared domains in proteins. If these domains are found in the same order, inferences on the function of the protein can be made.

The algorithm refined and reimplemented in this research, FIRE by Durand et al. (2010), is a global alignment algorithm which uses a modified Needleman-Wunsch algorithm. The algorithm requires time proportional to the product of the lengths of the aligned sequences, resulting in a time complexity of $O(mn)$ (Wang and Jiang, 1994). For two sequences: a and b, of m and n lengths respectively such that $a = a_1a_2...a_m$ and $b = b_1b_2...b_n$, the optimal alignment would be the alignment where the paired residues produce a maximum score. These scores can be arbitrary for nucleotide sequences or obtained from a substitution matrix for amino acid sequences.

The implementation of a global alignment approach which uses the Needleman-Wunsch algorithm, for example, the FIRE algorithm requires three basic steps: initialisation, iteration and traceback.

1. Initialisation

   This stage involves the creation of a matrix M which is an $(n+1)\times(m+1)$ matrix

where n and m are the sequence lengths. This matrix stores the scores of aligned residues. In the FIRE algorithm, three matrices are created for storing scores: *LEFT*, *UP* and *DIAGONAL*. These have corresponding matrices for storing the directions of alignments which are obtained from matrix M. Therefore, a total of six matrices are created in the FIRE algorithm. The number of matrices created is implementation dependent and influenced by such factors as performance and data structures available in the programming language.

2. Iteration

   In this stage, the algorithm finds all the scores for the possible alignments for sequences: a and b, this can also be done recursively. To find the optimal alignment, the sum of the residue scores is required. The iteration stage involves the filling of matrix M indexed by i and j created in the initialisation stage. Matrix M stores all the possible alignment scores for sequences: a and b, such that $M(i,j)$ is the score for (a_i, b_j) where i and j are indexes in $a_1...a_m$ and $b_1...b_n$. The matrix $M(i,j)$ is generated iteratively or recursively by:

$$M(i,j) = max \begin{cases} M(i-1,j-1) + s(a_i, b_j) \\ M(i-1,j) + g \\ M(i,j-1) + g, \end{cases} \tag{2.1}$$

   where $s(a_i, b_j)$ is the index of the substitution matrix for the aligned amino acids i and j and g is a constant gap penalty.

3. Traceback

   The final stage is the traceback which calculates the maximum match path using the matrix obtained from Equation (2.1). The summation starts from the last cell $M(m,n)$ in the *DIAGONAL* matrix and follows directions from either *UP* or *LEFT* matrices which were stored when building matrix M using indexes (i,j) such that $(i-1, j-1), (i-1, j)$ and $(i, j-1)$ correspond to matrices $DIAGONAL, LEFT$ and UP respectively. At each step in the traceback the alignment is built and gaps are inserted in the aligned sequences. The traceback

is complete when $i = 0$ and $j = 0$. There could be several optimal alignments for the two sequences; the choice of the alignment is algorithm dependent.

The requirement for faster algorithms has led to the Needleman and Wunsch (1970) algorithm seeing various improvements such as reduced memory usage in the Myers and Miller algorithm (Myers and Miller, 1988). This was subsequently implemented in the widely used CLUSTAL MSA program (Higgins and Sharp, 1988). Gotoh (1982) also worked on an implementation that reduced the number of computational steps. Faster global alignment methods have since been developed, for example, AVID (Bray, 2002), is a memory efficient program for comparing mega-bases of genomic regions. Notredame et al. (1998) developed the T-COFFEE program which utilises both the Needleman-Wunsch (Needleman and Wunsch, 1970) and the Smith-Waterman (Smith and Waterman, 1981). This resulted in relatively high CPU times for large data sets, however with improved accuracy.

The global alignment approach assumes that the sequences under study are similar across their whole length. Such an assumption may be flawed as genes may experience gene shuffling events (Raines, 1998); furthermore, in most cases the two sequences under investigation are not of equal length. Gene duplication events may also result in the rearrangement of putative regions. Moreover, the most common scenario is that only a subsection of one sequence is similar to the other sequence. On the other hand, in a study to benchmark the performance of sequence alignment algorithms using the BALIBASE Benchmark, Thompson and others found that global methods perform better than local alignment methods (Thompson et al., 2005). A global approach such as the Needleman-Wunsch algorithm can provide measures of the evolutionary distance and similarity between two sequences. Therefore, global algorithms are more suited for testing evolutionary hypothesis, this was the rationale that led to the FIRE algorithm being implemented using the Needleman-Wunsch algorithm.

2.1.2 Local alignment algorithms

Local alignment programs such as BLAST (Altschul et al., 1990) are implemented to rapidly search databases using the Smith and Waterman (1981) dynamic programming algorithm which finds common regions of similarity between aligned sequences. The Smith and Waterman (1981) algorithm was proposed as an alternative to the Needleman-Wunsch algorithm to produce local alignments instead of global alignments. Whilst the Smith-Waterman implementation is the most widely used, earlier implementations have also been found Sankoff (1972) and Sellers (1974). The algorithm maximises the similarity metric for a region of the aligned sequences to address some of the challenges faced by a global approach. Therefore, the local alignment approach is more effective at detecting functional domains such as the binding sites of transcription factors or regions where gene translocation (Raines, 1998) has occurred.

Vingron and Waterman (1994) provided a mathematical description for the local alignment of DNA sequences where a match is given an arbitrary score of 1 and a mismatch of μ was used, let two sequences be $a = a_1a_2...a_n$ and $b = b_1b_2...b_m$. If H_{ij} represents the elements of the matrix and these are calculated such that:

$$H_{ij} = max \begin{cases} H_{i-1,j-1} + 1, \text{if } a_i = b_j \\ H_{i-1,j-1} - \mu, \text{if } a_i \neq b_j \\ H_{i-1,j} - \delta \\ H_{i,j-1} - \delta \\ 0 \end{cases} \tag{2.2}$$

where δ is a constant gap penalty, which was abandoned for the affine gap (Gotoh, 1982) penalty to model the biological behaviour of indels, gap penalties are discussed in detail in Section 2.5. Equation (2.2) can be compared with Equation (2.1) where the main difference is the zero in Equation (2.2) in the Smith-Waterman implementation.

The Smith-Waterman algorithm tries to find the alignment that maximises the value of $H_{i,j}$ in the traceback stage, therefore the implementation bears some similarity to the global algorithm provided in Section 2.1.1.

The strength of the local alignment approach has been in database searching in which query sequences can be compared with database sequences to infer function (Bateman, 2004; Benson et al., 2010) or structure (Bernstein et al., 1977; Murzin et al., 1995). BLAST has become a ubiquitous database search tool. It utilises a heuristic approach based on the maximal segment pair (MSP) method, in which the algorithm generates short segments of alignments called "words" of a defined length which produces a *ktup* score (Altschul et al., 1990). The MSP score method and the stochastic properties of MSP scores (Karlin and Altschul, 1990) of the BLAST algorithm allow for the rapid analysis and assessment of the statistical significance of results. The challenge of statistical significance in sequence analysis is addressed in Section 2.7.

FASTA (***FAST*-*A***ll) by Pearson and Lipman (1988) was one of the first widely used sequence alignment program. It utilises a heuristic method of local sequence alignment which results in rapid analysis. Similar to the BLAST approach, FASTA makes use of the short segments of sequences called "words" to obtain matches using DP implemented using the Smith-Waterman algorithm (Smith and Waterman, 1981; Pearson and Lipman, 1988). The fasta file format is a legacy of the implementation of the FASTA program, which was also capable of evaluating the statistical significance of alignments produced (Pearson and Lipman, 1988). The SSearch algorithm also uses a modified form of the Smith and Waterman (1981) algorithm and generates alignments by finding regions of similarity between two sequences.

The local alignment approach has also been used in the comparisons of full genome

sequences, for example, BlastZ is an algorithm specialised for detecting local alignments in full genome sequences (Schwartz et al., 2003). Modifications to the local alignment algorithm with heuristics have also been proposed, DIALIGN (Morgenstern et al., 1998) and CHAOS (Brudno, Chapman, Göttgens, Batzoglou and Morgenstern, 2003) utilise improved modifications of the Smith-Waterman algorithm to determine optimal alignments.

2.2 Pair-wise alignment algorithms

Alignment algorithms can be used to compare two (pairwise alignments) or several (multiple sequence alignment (MSA)), sequences at a time. Local pairwise alignment methods such as BLAST (Altschul et al., 1990) and FASTA (Pearson and Lipman, 1988) are able to detect regions of translocation between two sequences where global alignment based algorithms are likely to fail. The ***F***lexible structure ***A***lignmen***T*** by ***C***haining ***A***ligned fragment pairs allowing ***T***wists (FATCAT) program generates alignments of flexible protein structures by pairwise comparison, the algorithm is well known for its accuracy (Ye and Godzik, 2003). It is important to note that methods such as FASTA and BLAST determine optimum alignments after the comparison of only a few of the total segments, however, these heuristic approaches have proved to be effective as these algorithms are able to produce accurate alignments rapidly.

The FIRE algorithm described and developed in this study is a pairwise global alignment algorithm implemented using a modified Needleman-Wunsch algorithm. It presents a novel approach to the traditional approach to sequence alignment where the similarity of ω MLEs profiles is used instead of amino acid residue similarities. The FIRE approach was conceptualised to perform alignments in what has become a challenging low sequence similarity region called the "twilight zone". The conceptual study on the FIRE algorithm provided evidence that supported the hypothesis that ω values at codon sites of a sequence may be used to infer protein domain function.

Using several datasets including orthologous and paralogous domains of low sequence similarity the study demonstrated that domains with similar ω MLEs values were under similar selective pressures and therefore responsible for the same function. The algorithm is therefore suited for detecting orthologous, paralogous or analogous genes responsible for similar functions at low sequence similarity. The algorithm compares the ω MLEs at codon sites from two MSAs of orthologous sequences and generates a pairwise alignment for two MSAs.

Three-way alignment algorithms have also been proposed for special cases such as the detection of homology in distant proteins (Doolittle, 1981). In an effort to align plastocyanin, stellacyanin, and cucumber basic blue proteins, Murata et al. (1985) developed a three-way sequence alignment algorithm called ALIGN3, based on the Needleman-Wunsch pairwise alignment algorithm.

2.3 Multiple sequence alignment

The FIRE algorithm compares two nucleotide MSAs of orthologs and generates a pairwise alignment of the amino acids using the spatial distribution of ω at codon sites. Durand et al. (2010) demonstrated that the performance of the FIRE algorithm compared well with widely used tools such as: FATCAT, T-COFFEE, CLUSTAL W and MAFFT for certain datasets. However, it is one of the objectives of this study to evaluate the new algorithm which is improved and re-implemented. Therefore, this section reviews some of the widely used algorithms and discusses the different approaches used in MSA, their implementations and challenges faced in developing them.

Multiple sequence alignments are more challenging to perform than pairwise sequence alignments since the running time scales with the lengths and number of sequences compared (Carrillo and Lipman, 1988; Wang and Jiang, 1994; Pei, 2008). To align N sequences of total length of L, the complexity is $O(L^N)$; the comparison of mul-

tiple sequences results in increased demands for computational resources (Wang and Jiang, 1994). For this reason the most widely used approach is a progressive alignment approach which constructs an MSA by successive applications of a pairwise alignment algorithm (Notredame, 2002), progressive alignment algorithms are discussed in Section 2.3.1. CLUSTAL (Higgins and Sharp, 1988) is the canonical MSA algorithm, it builds multiple alignments by progressively adding sequences using a heuristic approach. Alternative methods to address the challenge of MSA such as the consensus approach in the MULTAL program (Taylor, 1988) have also been proposed, although these methods have been found to be tractable, they are not very accurate. Genetic algorithms have also been provided in an attempt to address the challenge of comparing several sequences (Feng et al., 2002).

The ***M***ultiple ***A***lignment using ***F***ast ***F***ourier ***T***ransform (MAFFT) program is well known for its speed and accuracy in aligning large data sets (Katoh and Toh, 2008). The Fast Fourier Transform approach reduces the CPU time considerably by detecting similar regions in aligned sequences. In this novel approach amino acids are represented as frequencies of vectors representing the volume and polarity of the nucleotides (Katoh and Toh, 2008). A study evaluated a series of alignment tools and found that MAFFT performed the best at aligning protein sequences (Nuin et al., 2006). The success of the MAFFT algorithm has also been attributed to the speed at which it is able to generate accurate alignments from large data sets.

MUltiple ***S***equence ***C***omparison by ***L***og-***E***xpectation (MUSCLE) (Edgar, 2004) is also known for producing accurate alignments. The algorithm employs accurate sequence distance measuring methods called kmer distance and Kimura distance which results in highly accurate alignments (Edgar, 2004). MUSCLE and MAFFT perform as well as CLUSTAL W, in terms of alignment accuracy and they are able to generate these alignments at high speeds an important factor when comparing large data sets.

Other algorithms such as LAGAN (***L***imited ***A***rea ***G***lobal ***A***lignment of ***N***ucleotides) and Multi-LAGAN (Brudno, Do, Cooper, Kim, Davydov, Green, Sidow, Batzoglou et al., 2003) are more suited for high throughput projects such as the comparison of genomic DNA.

Tree based ***C***onsistency based ***O***bjective ***F***unction ***F***or alignm***E***nt ***E***valuation (T-COFFEE) developed by Di Tommaso et al. (2011) uses a library of sequences to generate an alignment and this library can be generated using other approaches, for example, CLUSTAL (Higgins and Sharp, 1988) or LAlign (Brudno, Do, Cooper, Kim, Davydov, Green, Sidow, Batzoglou et al., 2003). The consistency objective function which evaluates alignments produced ensures accurate alignments are produced at the cost of speed.

Measures of MSA have been reviewed by Gonnet et al. (2000) and the evaluation of several MSA programs for proteins has been studied by Nuin et al. (2006), however, there is no consensus on the best overall MSA approach. The increase in the numbers of alignment algorithms necessitated the conception of the first custom made benchmark test set for MSA programs called BAliBASE (Thompson et al., 2005). Other benchmark approaches have also been suggested, for example, SABmark (**S**equence **A**lignment **B**ench**mark**) is compiled from challenging alignments of sequences of low similarity in the twilight zone and superfamilies from a structural protein database called SCOP (Van Walle et al., 2005). APDB (**A**nalyze alignments with **PDB**) (O'Sullivan et al., 2003) uses the structure of proteins as a benchmark measure which removes the requirement of reference sequences. Similar to SABmark, HOMSTRAD is a database of structures of homologous protein family alignments which can be used to evaluate MSA tools (Mizuguchi et al., 1998).

2.3.1 Progressive alignment algorithms

The challenge of computational power and memory requirements has led to the innovative conceptualisation of a plethora of heuristic MSA approaches requiring less expensive computational times (Carrillo and Lipman, 1988). Among these approaches was the progressive MSA approach referred to as the hierarchical or tree method in some literature, first proposed by Feng and Doolittle (1987), however some earlier proposals have been found (Hogeweg and Hesper, 1984). To compare n sequences, let n be the number of sequences to be aligned; the number of pairwise alignments required would be $n \times (n-1)/2$. The results of the pairwise alignments are then used to calculate a phylogenetic tree called the guide tree, if one is not provided. This guide tree determines the order in which sequences are added to the final alignment. Guide trees can be generated separately by algorithms for estimating phylogenetic trees such as the UPGMA (Sneath et al., 1973) or the NJ algorithm (Saitou and Nei, 1987), these algorithms are discussed in Section 2.8. The progressive approach has been described as "greedy" as the alignments generated at each step are not necessarily the correct ones. These are retained until the final alignment; the order in which the sequences are compared affects the final alignment.

CLUSTAL W (Thompson et al., 1994) (W for weights) is a variant of the CLUSTAL algorithm (Higgins and Sharp, 1988), it is the most widely used MSA algorithm. The algorithm is based on a progressive alignment algorithm for closely related sequences. In CLUSTAL W and other progressive approached algorithms alignments are generated by adding closely related sequences using a specified guide tree. The alignment is maintained, therefore, any errors earlier on will affect the final alignment, which is a weakness. CLUSTAL Omega is another variant of the CLUSTAL program based on the progressive alignment heuristic (Sievers et al., 2014). It produces better alignments than the CLUSTAL W with better execution times. This increase in performance was achieved by the adoption of a faster guide tree generating algorithm. This modification

has reduced the time complexity of the guide tree generation from $O(N^2)$ to $O(NLogN)$ resulting in faster alignment generation times. Additionally, accurate alignments are generated by a hidden Markov model program called HHalign by (Söding, 2005). The CLUSTAL Omega algorithm has demonstrated superior performance on large numbers of benchmark sequences, however, it does not perform as well as MAFFT in terms of quality of alignment.

On the other hand, some MSA tools make use of a consistency objective function, for example, T-COFFEE (Di Tommaso et al., 2011) a member of the COFFEE (Notredame et al., 1998) family of algorithms is a successful attempt at increasing the accuracy of MSAs. It is also based on a progressive approach with modifications to avoid some inconsistencies associated with the computational greediness of the approach. Consequently, the T-COFFEE algorithm is able to make very accurate alignments of very divergent proteins but only for small sets of sequences, given its high computational requirements. Other algorithms making use of the objective function to increase accuracy at the cost of performance include: MSAprobs (Liu et al., 2010), Probalign (Roshan and Livesay, 2006) and Probcons (Do et al., 2005).

Progressive algorithms do not produce optimal alignments; due to the greedy nature of the algorithm, where early mistakes in the alignment cannot be corrected. Iterative methods discussed in the next section and probabilistic approaches in Section 2.3.3 provide useful alternatives to the progressive approach.

2.3.2 Iterative methods of alignment

Iterative approaches generate alignments by iteratively performing pairwise alignments on random groups of sequences until an optimal alignment is produced (Gotoh, 1995). The iterative approach avoids the greedy and error prone limitations of the progressive alignment approach in which errors in the initial alignment are retained, affecting the final alignment (Berger and Munson, 1991; Corpet, 1988). Hirosawa et al. (1995)

provides an in depth investigation of iterative algorithms. Gotoh (1996) also developed the PRRP algorithm which generates alignments iteratively based on a hill-climbing algorithm.

The MUSCLE algorithm, popular for its accuracy generates alignments based on an iterative approach (Edgar, 2004). Other iterative approach implementations include DIALIGN (Morgenstern et al., 1998), MAFFT and DALIGN where segment to segment comparisons based on local alignments are employed. On the other hand in MAFFT, pairwise alignments produce a Weighted Sum of Pairs (WSP) score; this heuristic approach results in rapid analyses with MAFFT being faster than CLUSTAL W.

In a paper comparing MSA algorithms Thompson et al. (1999) failed to reach a conclusion on the improvements brought about by iterative methods over progressive heuristic methods in terms of quality of alignments produced. They however, discovered that iterative methods had better alignment accuracy and that such methods produced better alignments when provided with more information. In their comparative study Thompson et al. (1999) also found that DIALIGN was the most successful iterative method.

2.3.3 Probabilistic approaches to sequence alignment

Maximum Likelihood and Bayesian methods are increasingly being used as alternatives to cost based algorithms. These methods use probabilistic based approaches to model mutational events to generate alignments (Thorne et al., 1991). Bayesian approaches to determine accurate sequence alignments and phylogeny generation have also been provided, for example, Lunter et al. (2005) provide a method for determining phylogeny and generating alignments based on a Bayesian Markov chain Monte Carlo approach.

Recently, Hidden Markov Models (HMM) have become widely accepted for modelling insertions and deletions in alignments (Allison et al., 1992; Holmes and Bruno,

2001), early work in this regard was carried out by Baldi et al. (1994) and Krogh et al. (1994). The HMMER program (Finn et al., 2011) uses HMMs to generate alignments; the algorithm is used to classify proteins into domain families in the PFAM database (Bateman, 2004). The ProbCons algorithm by Do et al. (2005), generates multiple alignments by progressively applying pairwise HMMs alignments to sequences. Additionally, the ProbCons algorithm utilises a probabilistic consistency function which has resulted in qualities of alignments better than popular tools such as CLUSTAL W (Do et al., 2005). HMMs have also been used to detect relatedness between proteins, for example, in the SAMT98 algorithm (Karplus et al., 1998). The success of probabilistic approaches, for example, HMMER in clustering proteins in the PFAM database has increased the acceptance of probabilistic based methods in sequence analysis. In addition some tools have been developed to accept HMM profiles as input to generate alignments, for example, CLUSTAL Omega (Sievers et al., 2014). However, the challenge is that probabilistic based algorithms are complex, not well understood and computationally expensive.

2.4 Substitution matrices

One of the most crucial parameters of a protein sequence alignment algorithm is the substitution matrix also referred to as similarity or scoring matrices in the literature. They score residues aligned in two amino acid sequences, these scores are then used to find the optimum alignment (Altschul, 1991). Several of these matrices have been provided (Dayhoff and Schwartz, 1978; Henikoff and Henikoff, 1992; Overington et al., 1992; Jones, 1999; Gonnet and Korostensky, 1999); their use can be traced back to the introduction of such algorithms as the Needleman-Wunsch Algorithm (Needleman and Wunsch, 1970). Numerous methods for developing a matrix to fit the compositional bias of the sequences have been proposed. On the other hand, the most common approaches are to either adapt an existing matrix to the data or to build a new matrix empirically from the data. The main objective of this study is to conceptualize an

approach to improve and re-implement a new version of the algorithm, where alignments are generated using the ω at codon sites as a similarity metric. The objective will involve increasing the specificity of the algorithm and the efficacy of incorporating substitution matrix scores for aligned residues will be evaluated.

2.4.1 The PAM matrices

The PAM (Point Accepted Mutation or Percentage Accepted Mutations in some literature) represent the first protein substitution matrices to be widely used in sequence analysis (Dayhoff and Schwartz, 1978). They are derived from evolutionary models to produce a 20×20 matrix of log-odd scores which were extrapolated to produce the PAM family of matrices. The PAM matrices were extrapolated using sequences with a minimum of 85% sequence identity, extrapolation was utilised to generate other matrices of different evolutionary distances as at the time there was limited sequence information available. However, some of the assumptions on which the PAM matrices are based on have been questioned (Bennet et al., 1994).

Let PAMn represent a member of the PAM family of matrices, where n represents the number of accepted point mutations per 100 amino acids. Therefore, n also indicates the matrix derived from multiplying the PAM matrix by itself n times, for example, to derive the PAM250 the PAM1 is multiplied by itself 250 times. Let (i, j) represent an index of an element on the PAM matrix given by M, then the score of each entry, $s(i, j)$, is given by:

$$s(i,j) = log\frac{M^n(i,j)}{p(i)p(j)} \tag{2.3}$$

where $p(i)$ and $p(j)$ are the observed frequencies of amino acid i and j, M is the PAM matrix and n is the number of extrapolated point mutations. Therefore, the likelihood ratio or PAM score for a mutation from i to j is calculated using the ratio of the probability that the change is a mutation and the probability that the change is by chance as provided in Equation (2.3). Dayhoff and colleagues assumed that the

frequencies of amino acids are constant over time and that one point accepted mutation produced the same mutations over longer periods of time, a concept similar to the molecular clock hypothesis (Zuckerkandl and Pauling, 1965). For any aligned amino acid residues, a PAM score greater than zero indicates a likely mutation, a value equal to zero indicates a random mutation and scores less than zero indicate a statistically unlikely mutation. The general rule is to use low PAM matrices for local similarities and the high PAM numbers for finding long weak similarities. The PET91 matrix is an updated version of the PAM matrices where instead of extrapolated log-odds empirical data were used to develop the matrices (Jones et al., 1992). The general approach is to use the PAM matrices for global alignments and BLOSUM matrices for local alignments, however, the biological context of alignments must be taken into account.

2.4.2 The BLOSUM matrices

The BLOSUM matrices (***BLO***cks ***SU***bstitution ***M***atrix) have become the default scoring matrices for powerful multiple sequence analysis tools such as CLUSTAL W (Thompson et al., 1994) and database search tools like BLAST (Altschul et al., 1990). These matrices are log-odd scores of matches and mismatches constructed empirically from blocks of sequences of particular divergence (Henikoff and Henikoff, 1992). Each score in a BLOSUM matrix, for example, in the BLOSUM62 matrix is the bit score of the expected frequencies and observed frequencies for two aligned amino acids. The BLOSUM family of matrices have been found to be superior in detecting homologs when compared to the PAM matrices. This has been attributed to the fact that BLOSUM matrices were constructed using real data and the PAM matrices were extrapolated using evolutionary distances from a small data set (Henikoff and Henikoff, 1992, 1993). The BLOSUM score for amino acids i and j can be calculated as follows:

$$S_{ij} = \left(\frac{1}{\lambda}\right) log_2 \left(\frac{p_{ij}}{q_i \times q_j}\right) \tag{2.4}$$

where p_{ij} is the observed frequency of i and j replacing each other, and q_i and q_j are the expected frequencies of i and j in any protein, λ is a scaling factor which converts

the bit score to integers which are then rounded off (Henikoff and Henikoff, 1992). Comparatively, Equation (2.3) and Equation (2.4) bear some similarity and it was the source of data used that resulted in different performances.

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V	B	Z	X	*
A	4	-1	-2	-2	0	-1	-1	0	-2	-1	-1	-1	-1	-2	-1	1	0	-3	-2	0	-2	-1	0	-4
R	-1	5	0	-2	-3	1	0	-2	0	-3	-2	2	-1	-3	-2	-1	-1	-3	-2	-3	-1	0	-1	-4
N	-2	0	6	1	-3	0	0	0	1	-3	-3	0	-2	-3	-2	1	0	-4	-2	-3	3	0	-1	-4
D	-2	-2	1	6	-3	0	2	-1	-1	-3	-4	-1	-3	-3	-1	0	-1	-4	-3	-3	4	1	-1	-4
C	0	-3	-3	-3	9	-3	-4	-3	-3	-1	-1	-3	-1	-2	-3	-1	-1	-2	-2	-1	-3	-3	-2	-4
Q	-1	1	0	0	-3	5	2	-2	0	-3	-2	1	0	-3	-1	0	-1	-2	-1	-2	0	3	-1	-4
E	-1	0	0	2	-4	2	5	-2	0	-3	-3	1	-2	-3	-1	0	-1	-3	-2	-2	1	4	-1	-4
G	0	-2	0	-1	-3	-2	-2	6	-2	-4	-4	-2	-3	-3	-2	0	-2	-2	-3	-3	-1	-2	-1	-4
H	-2	0	1	-1	-3	0	0	-2	8	-3	-3	-1	-2	-1	-2	-1	-2	-2	2	-3	0	0	-1	-4
I	-1	-3	-3	-3	-1	-3	-3	-4	-3	4	2	-3	1	0	-3	-2	-1	-3	-1	3	-3	-3	-1	-4
L	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4	-2	2	0	-3	-2	-1	-2	-1	1	-4	-3	-1	-4
K	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5	-1	-3	-1	0	-1	-3	-2	-2	0	1	-1	-4
M	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5	0	-2	-1	-1	-1	-1	1	-3	-1	-1	-4
F	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6	-4	-2	-2	1	3	-1	-3	-3	-1	-4
P	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7	-1	-1	-4	-3	-2	-2	-1	-2	-4
S	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4	1	-3	-2	-2	0	0	0	-4
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5	-2	-2	0	-1	-1	0	-4
W	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11	2	-3	-4	-3	-2	-4
Y	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	-1	-3	-2	-1	-4
V	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4	-3	-2	-1	-4
B	-2	-1	3	4	-3	0	1	-1	0	-3	-4	0	-3	-3	-2	0	-1	-4	-3	-3	4	1	-1	-4
Z	-1	0	0	1	-3	3	4	-2	0	-3	-3	1	-1	-3	-1	0	-1	-3	-2	-2	1	4	-1	-4
X	0	-1	-1	-1	-2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-2	0	0	-2	-1	-1	-1	-1	-1	-4
*	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	1

Figure 2.1: The BLOSUM62 substitution matrix, taken from the European Molecular Biology Open Software Suite (EMBOSS) software by Rice et al. (2000). The top row and first column on the left represents amino acid characters, the diagonal scores are the identical match scores for aligned amino acid residues.

The BLOSUM matrices were empirically compiled from alignments extracted from the BLOCKS database which is a repository of protein motifs extracted from the PROSITE database of motifs (Falquet et al., 2002). In addition, since the BLOSUM matrices were developed using regions of conserved sequence similarities; the frequencies used in calculating scores reflect functional or structural constraints (Henikoff and Henikoff, 1992). In an experiment to evaluate the performance of the MAFFT algorithm in generating alignments, Katoh and Toh (2008) showed that BLOSUM62 (Figure 2.1) matrix is the best of the BLOSUM family of matrices. Furthermore, the

BLOSUM62 is the default scoring matrix for the BLAST algorithm, a change from PAM-120 matrix reflecting the superiority of BLOSUM matrices. In comparison with BLOSUM, PAM matrices have also been found to be negatively correlated to the BLOSUM matrices, such that the BLOSUM62 is roughly equivalent to the PAM120 matrix.

Gonnet and Korostensky (1999) proposed the GONNET scoring matrix for the frequency of substitutions in consecutive amino acid pairs, it is based on the work by Dayhoff on PAM, however, they are not widely used. Other more specialised matrices been suggested, for example, for Mollecutes bacterium analysis (Lemaitre et al., 2011), detection of homologs in low complexity glycoregions in *Saccharomyces cerevisiae* (Coronado et al., 2006) and for Plasmodium protein comparative studies (Brick and Pizzi, 2008). Lemaitre et al. (2011) point out that sequence compositions vary from organism to organism and from the source of the sequence for example the transmembrane proteins. Studies on the effect of the composition of amino acids substitution matrices on the results of a sequence alignment highlight the effects of choice of substitution matrix on the identification of homologs (Coronado et al., 2006; Brick and Pizzi, 2008; Paila et al., 2008). Lemaitre et al. (2011) addressed the challenge of divergence in biological sequence composition in species in which conventional matrices such as BLOSUM do not apply and produced a substitution matrix especially for Molluces bacteria AT-rich genomes.

The use of PSSMs (Position Specific Scoring Matrices) is also specialised, for example, PSI-BLAST (Altschul et al., 1997) uses a position-specific scoring matrix. In a PSSM a matrix of a multiple alignment is constructed by using the position of the residues in the alignment and specifying the frequencies of the amino acids in each position in each column. The frequencies for each of the 20 amino acids are calculated using the aligned sequences which are weighted by position-based weights. PSSMs are used to construct sequences which can be used for algorithms for comparing single se-

quences instead of multiples. PSSM multiple alignments have also been used to search databases (Gribskov et al., 1987).

Recently, the efficacy of substitution matrices has been questioned with the development of novel search approaches that do not utilise amino acid substitution matrices. For example, the CS-BLAST tool, is a novel search approach in which context-specific (CS) substitution matrices were incorporated into the BLAST algorithm (Angermüller et al., 2012). At the same time, MIQS (**M**atrix to **I**mprove **Q**uality in **S**imilarity search), a more robust matrix, has been developed from numerous matrices using principal component analysis for detecting remote homologies in transmembrane regions (Yamada and Tomii, 2013).

2.5 Gap penalties

Mutational events such as insertions or deletions make the challenge of generating alignments in some cases computationally intractable. To address this challenge, gap penalties are implemented in the algorithm to approximate insertions and deletions in the aligned sequences. The implementation of a gap penalty scheme has been an important consideration since the inception of the first sequence alignment algorithms (Needleman and Wunsch, 1970; Bernstein et al., 1977; Smith and Waterman, 1981). However, penalty parameter choice still remains a poorly understood concept with most researchers using the default gap open and extension penalty parameters provided by algorithms. Vingron and Waterman (1994) reviewed the effects of parameter choice on alignments and identified a common error that researchers made was to confuse different algorithm parameters for new algorithms. In this section some of the common penalty schemes and the scheme implemented in the FIRE algorithm are discussed.

The most common strategy for the insertion of gaps is the affine gap penalty proposed by Gotoh (1982). It is in the following form: $P = g + el$, where P represents the

total affine gap penalty, g is the cost of initiating a gap, e is the gap extension penalty and l is the length of the gap (Gotoh, 1982; Altschul and Erickson, 1986). This penalty scheme tries to model the biological behaviour of sequences where insertions and deletions are more likely to span multiple residue sites. Therefore, the scheme does not penalise the extension of gaps at the same cost of gap initiation. The gap costs are such that the gap open penalty is considerably more than the gap extension penalty. Generally, these gap parameters can usually be varied as an option of the algorithm in use. Studies have been carried out to explore the effects of varying the gap open and extension penalties in ribosomal data sets (Bernstein et al., 1977; Carroll et al., 2006) and using position specific gap penalties to improve MSA (Thompson et al., 1994).

The original Needleman-Wunsch algorithm used a constant gap penalty approach to produce alignments which has since lost popularity to the affine gap model. Barton and Sternberg (1987) also found evidence suggesting that length dependent gap penalties were not essential, especially in the case where large gaps are involved as observed by Murata et al. (1985) in haemoglobin. Barton and Sternberg (1987) argue for gap penalties weighted for residues in secondary structures to improve the quality of the alignments. Gap costs may also be implemented for terminal residue positions and these can result in optimal alignments with fewer internal gaps. The three-sequence algorithm proposed by Murata et al. (1985) utilised a constant gap penalty where the cost of the gap was independent of the length.

The CLUSTAL W algorithm utilises a variable gap penalty approach determined by such factors as the length of sequences, similarity of sequences and divergence in sequence length (Thompson et al., 1994). In the FIRE algorithm an affine gap penalty scheme was also adopted. An interesting area of study is the relationship between the match scores and the gap cost, this is mostly in DNA where a single value can be used for a match score or a mismatch score. However, in the case of proteins where

empirically determined substitution matrices are used, determining such a relationship is not trivial.

Generally high gap penalties result in fewer gaps in alignments. On the other hand, fewer gap penalties result in more gaps in alignments and more residue matches therefore the choice of these parameters must always be made in a biological context. This means a different strategy is adopted to align distantly related sequences in the twilight zone; lower gap penalties should be adopted. Conversely, for closely related sequences high gap penalties should be adopted. Empirical studies on determining optimal gap parameters are required to guide users on the selection of biologically optimum parameters (Reese and Pearson, 2002; Carroll et al., 2006). It has also been observed that most studies providing algorithms do not provide guidance on the selection of these parameters.

2.6 The quality of sequence alignments

The quality of an alignment is sometimes evaluated using the percentage identity (PID) of the pairwise alignment of proteins or nucleotides. Decisions on homology or shared functions may be inferred from the PID. On the other hand, the calculation of this metric is rarely the focus of any studies. In an effort to shed light on this topic, Raghava and Barton (2006) studied the variance of the PID score due to the variation in calculation methods. They found that the maximum variation in percentage due to the difference in calculation method was 11.5 % and the effect of the algorithm was found to be 14.8%. These results and other studies (Agrawal and Huang, 2009) confirm what has become common knowledge, that the percentage identity (PID) is a poor indicator of sequence similarity. Adding to the confusion is the failure for researchers to distinguish sequence similarity, which compares the physiochemical similarities between amino acids from sequence identity where matched residues are counted. Calculating this metric varies, some approaches use the percentage of the ratio of the total number

of matched residues and the length of the shortest sequences, others use the longest sequence to normalise the score and others use the mean sequence length (Barton and Sternberg, 1987). However, in distantly related sequences in the twilight zone using the length of the aligned positions with the unaligned region is excluded may be more appropriate (Doolittle, 1981). Therefore, a combination of the PID score and some other information must be used to infer shared functions and possibly homology.

2.7 The statistical significance of alignments

A very common task in sequence analysis is querying a sequence database like GenBank (Altschul et al., 1990) with an algorithm such as BLAST (Altschul et al., 1990) using a DNA sequence. Such a query will yield results including the score of the top alignments and it now falls on the researcher to determine which of these results can be used to infer the relationship between sequences. Therefore, sequence alignment algorithms require statistical frameworks to assess the significance of alignments, for example, in pairwise algorithms such as FASTA (Pearson and Lipman, 1988) and in database search algorithm such as BLAST to distinguish between real alignments and those due to chance. The requirement for the ability to evaluate the significance of database search result has become an important aspect of algorithm design as biological databases increase in size.

Most sequence alignment algorithms will provide a score for the analysis, in databases this is usually a list of hits and their scores. Inferences of shared function and homology can then be made using this score. However, the decision of homology should never be made based only on the alignment score, it is general practice to also assess the statistical significance of an alignment. Phylogenetic analyses are then used to confirm homologous relationships. The challenges of using alignment scores such as the PID to infer the quality of an alignment are discussed in Section 2.6. Some early approaches of assessing the statistical significance of alignments include the work of Karlin and

Altschul (1990) and Pearson (1998). If the probability of getting an alignment by chance is rare, then that alignment is good. This provides the general framework for the evaluation of the statistical significance of an alignment. Consequently, the statistical significance of an alignment is a better indicator of homology than the score of an alignment (Agrawal and Huang, 2009). Algorithms such as SSearch can be utilised to search for a sequence similar to a query sequence (Smith and Waterman, 1981). It utilises a statistical method to identify those sequences that are not similar by chance, this is achieved by using a regression test and the natural log of sequences to generate Z scores for the aligned sequences. The solution to the statistical significance challenge is dependent on such factors as the type of alignment produced: local or global alignments, gapped or ungapped alignments. Agrawal and Huang (2009) propose a multi-parametric measure to determine the statistical significance of an alignment and show that such an approach is more robust than the conventional single parameter.

It is important not to confuse the biological significance and the statistical significance of an alignment, sequences may not be significantly related because they are distantly related in the "twilight zone" and the effect of other factors such as the scoring matrix and the objective function must not be overlooked (Agrawal and Huang, 2009). The final truth of the biological significance and relationships inferred can only be proved through experimentation or through structural determination methods (Bernstein et al., 1977; Murzin et al., 1995; Brenner et al., 1998). The goal of this study is to use a new and improved algorithm to search a database to infer domain function of novel proteins. This will require a statistical framework to assess the quality of alignments obtained.

2.7.1 The *p*-value and E-value

Generally, the statistical significance of alignments is assessed using the p-value. The concept of using the p-values has its roots in hypothesis testing where the goal is to

determine which of two hypotheses, the null hypothesis or the alternative hypothesis is true. In the case of alignments the test statistic is the score of the alignment. The null hypothesis is that the two sequences are related by chance and the alternative, hypothesis is that they are indeed related. Therefore, the p-value is the probability of accepting a false null hypothesis or accepting that sequences are not related. Parametric procedures can be used to determine the p-value, which involves fitting a distribution, for example, Poisson, Normal or extreme value distribution to scores produced to calculate the p-value. One of the challenges of dealing with alignment scores in parametric methods, is that they violate the assumption of normal distribution. However, local alignment scores have been found to follow an extreme value distribution with a Gumbel shape and this has been used to develop a statistical assessment framework (Karlin and Altschul, 1990). Global alignment scores on the other hand, are more challenging to deal with and a p-value approach is sometimes adopted.

The p-value of a database search is the probability that the alignment score is due to chance. This is the probability of obtaining a score equal to or higher than the alignment score by chance. To compute the p-value, a model for generating scores must be chosen. The most common model for generating the p-value is to use a random sequence. The random sequence can be real, non-homologous or sequences that have been generated randomly based upon a coding sequence model or some amino acid distribution. Another approach of generating randomness is to select random sequences from a database as seen in Pearson (1998) and Waterman and Vingron (1994), the rationale is that real sequences provide the best data. Historically, real shuffled sequences were utilised to conserve the compositional properties of the sequence. Shuffling a local sequence preserves the probability distribution of the amino acids. Therefore, the simplest approach to assessing the statistical significance of an alignment would be to generate random alignments and assume a normal distribution of the alignment scores. If the alignment score is x, then the probability $P(s >= x)$ will be the proportion

of shuffled sequences scoring above s, while such an implementation is trivial, it is computationally expensive.

The E-value (Expectation value) is another metric which has gained wide spread use. The E-value has been popularised by the BLAST algorithm as an indicator of the quality of an alignment (Altschul et al., 1990). It indicates the relatedness of the query sequence and a database sequence. To distinguish those alignments due to chance and those that can be used to infer homology, the E-value is assessed. It is used as an indicator of how many times the alignment score can be expected by chance alone. Therefore, the E-value is more intuitive than the p-value as it is also correlated to the length of the query sequence and database length.

2.8 Phylogenetic trees

Phylogenetic trees represent a hypothesis of evolutionary relationships between operational taxonomic units (OTUs). These OTUs are context dependent and in molecular biology they can be DNA sequences, RNA or protein sequences. In this study, DNA and protein sequences will be discussed. Several approaches to phylogenetic tree generation have been proposed; early implementations of the clustering algorithms used parsimony based methods to infer phylogeny (Sankoff, 1975) these included using weighted averaging (Waterman and Perlwitz, 1984) and the weighted least square approach also known as the Fitch-Margoliash method (Fitch et al., 1967).

Phylogenetic relationships are represented using rooted and unrooted binary trees, where leaves represent sequences and the nodes are bifurcations representing hypothetical ancestral units. In the phylogenetic tree the length of the edge represents the similarity or edit distance between the sequences. With the advent of the molecular clock hypothesis (Zuckerkandl and Pauling, 1965), the branch length has also been interpreted to represent the time since their divergence. Phylogenetic construction can

be either distance or character based.

Distance based approaches make use of the distances between the taxonomic units such as sequences. Various approaches for determining the distance between sequences have been proposed. For two sequences, the simplest method of calculating distance is to use the fraction of residue sites where the sequences differ, Jukes et al. (1969) provide a method based on this simple approach. Character based methods on the other hand use models of evolution and try to determine the most optimal tree that describes the ancestry of the sequences, these can be maximum parsimony or maximum likelihood (ML) based. The ML based approaches have become more popular and widely used in phylogenetic tools. While several clustering algorithms for producing phylogenetic trees have been proposed, the two algorithms: the Unweighted Pair-Group Method with Arithmetic mean (UPGMA) by Sneath et al. (1973) and the Neighbour-Joining (NJ) Method (Saitou and Nei, 1987; Studier and Keppler, 1988) are used as clustering algorithms in phylogenetic tools.

The FIRE algorithm requires ω at codon sites to produce alignments, however, to determine ω, nucleic acid sequences and phylogenetic trees are required. In the FIRE concept paper, CLUSTAL W (Thompson et al., 1994) and PAUP (Swofford, 2003) tools were used to construct trees. In this study phylogenetic trees for the PFAM database will be required to calculate ω at codon sites for the PFAM families. Therefore, in the next sections, some of the most widely used phylogenetic tree drawing algorithms and approaches are described and discussed.

2.8.1 The UPGMA algorithm

The ***U***nweighted ***P***air ***G***roup ***M***ethod using ***A***rithmetic mean (UPGMA) method pioneered by Sokal and Michener (1958) and refined by Sneath et al. (1973) also known as the average linkage method is a greedy clustering algorithm for generating phylogenetic trees, where trees are constructed by recursively joining and grouping the closest

nodes into one cluster. This hierarchical agglomerative clustering algorithm finds the phylogenetic tree by joining proximal OTUs to create nodes on the phylogenetic tree. The algorithm produces a tree where the distance between the nodes to any one of its leaves is the same. It also produces trees based on the molecular clock assumption (Zuckerkandl and Pauling, 1965).

The UPGMA algorithm is a popular clustering algorithm because it is easy to implement and computationally cheap with relatively low memory requirements (Katoh and Toh, 2008). One of the important factors for choice in clustering algorithms for inferring phylogenetic trees is the ability to handle large matrices, this was one of the considerations for the creators of the first version of the CLUSTAL algorithm (Higgins and Sharp, 1988). In their study, Higgins and Sharp (1988), highlight that the accuracy of a tree generated by the UPGMA method will depend on whether the assumption of constant mutation rates has been violated. In the MAFFT algorithm guide trees are constructed using the UPGMA method (Katoh and Toh, 2008). Similarly, in the Evolutionary Trace (ET) algorithm sequences are clustered using the UPGMA method to find a consensus sequence whose regions are then marked according to conservation; the status of each site is integrated in the elucidation of the three dimensional structure (Lichtarge et al., 1996).

Researchers have attempted to evaluate the efficacies of these methods as early as in the 1980s to investigate the performance of various algorithms in estimating tree topologies of given data. Nei et al. (1983) found that the UPGMA and FM methods were more efficient than other available methods at the time in obtaining more accurate rooted trees at constant substitution rates given large numbers of nucleotide substitutions.

2.8.2 The NJ algorithm

The ***N***eighbour-***J***oining (NJ) algorithm was developed to overcome some of the shortcomings of the UPGMA such as clustering sequences where the mutation rates are not uniform (Saitou and Nei, 1987). In the NJ method, a distance matrix is used to account for the variation in divergence rates so that proximal nodes, but far from other nodes are not joined together. It produces unrooted trees unlike the UPGMA method and it does not assume a molecular clock hypothesis and the method is reliable when the lengths are allowed to vary. The NJ method has been found to be most effective when more than 20 sequences are being analysed to construct a phylogenetic tree (Saitou and Nei, 1987).

Early CLUSTAL versions used the UPGMA method to construct guide trees however the CLUSTAL W version (Thompson et al., 1994) uses the NJ algorithm. Thompson et al. (1994) supported this change in phylogenetic construction algorithm from the UPGMA to the NJ method citing that the method allowed for variation in the evolutionary rates which resulted in more accurate estimates of branch by branch lengths. Furthermore, the CLUSTAL W method requires accurate branch lengths to determine sequence weights and experiments with SH3 domain have revealed that these and other modifications resulted in increased algorithm sensitivity (Thompson et al., 1994).

Datamonkey (Delport et al., 2010) is a webserver for phylogenetic analysis and generates phylogenetic trees based on the NJ method using Tamura and Nei distances (Tajima and Nei, 1984). The Rate4Site algorithm on the other hand, determines functionally important sites using the evolutionary rate and structural information, the algorithm uses the NJ method to construct phylogenetic trees (Pupko et al., 2002). Finally, the FastTree (Price et al., 2010) algorithm was implemented using a combination of the NJ method and a heuristic approach to calculate the phylogenetic trees rapidly based on a ML approach, the algorithm is also used to cluster proteins into families

in the PFAM database. Phylogenetic trees for the PFAM database will be required to determine the evolutionary rate profiles for the database to infer domain functions of proteins.

2.8.3 Maximum parsimony

Maximum parsimony methods also referred to in some literature as the minimum evolution method attempt to produce a tree that minimises the number of steps required to generate the observed sequences (Fitch, 1971). This approach has been found to bear some similarity or may have been inspired by the principle of Occam's razor. Comparatively, the NJ algorithm has been found to be a less complex implementation of the maximum parsimony method (Saitou and Nei, 1987; Studier and Keppler, 1988). In maximum parsimony the topology of the phylogenetic tree chosen is the one with fewest evolutionary changes independent of the number of sequences (Rzhetsky and Nei, 1995). The advantages of parsimony methods include their simplicity and their intuitive implementation. The method produces a tree that provides an explicit model, however, Yang and Rannala (2012) point out that the method has implicit assumptions and the method is not well understood. Early criticisms of this approach have also been found (Felsenstein, 1978), in addition to these criticisms is that since maximum parsimony is based on some model of evolution then *a proiri* knowledge of the sequences cannot be incorporated.

A plethora of tools, for example, PHYLIP (Felsenstein, 2006) and PAUP (Swofford, 2003) provide parsimony based methods. MEGA, one of the widely used tools, is a suite of software for phylogenetic analysis, used in the inference of phylogenetic relationships and testing hypothesis in molecular evolution studies (Kumar et al., 2004). It provides maximum parsimony methods and distance based methods such as UPGMA, NJ and minimum evolution for phylogenetic cluster constructions.

2.8.4 Maximum likelihood and Bayesian based approaches

Cavalli-Sforza and Edwards (1967) introduced likelihood methods to phylogenetic inference, however, it was until the framework provided by Felsenstein in 1973 that these methods could be utilised (Felsenstein, 1973). This framework for the application of the maximum likelihood method led to the earliest application of likelihood estimation to phylogenetic tree construction. Maximum Likelihood methods attempt to find a tree that best models the variation of the observed sequences. These methods are based on several assumptions such as the independent evolution of characters independently, constant mutation rates and the molecular clock (Huelsenbeck and Crandall, 1997). Likelihood methods are powerful tools for testing models and hypothesis and the ability to use models of evolution means they are more biologically realistic (Yang and Rannala, 2012).

Maximum likelihood phylogenetic inference tools such as PhyML (Guindon et al., 2010) and RAxML (Stamatakis, 2006) are widely used for generating phylogenetic trees. On the other hand, to cope with the large amounts of sequence data becoming available, faster ML based tree calculating algorithms have been developed, for example, FastTree (Price et al., 2010). The FastTree algorithm uses a heuristic approach combined with the ML approach to generate good quality tree topologies for thousands of sequences. A popular implementation of the maximum likelihood method is the Phylogenetic Analysis by Maximum Likelihood (PAML) program (Yang, 2007). Yang (1996) evaluated the efficacy of parsimony and likelihood based methods. In this study with simulated data, experiments revealed that the performance of parsimony based methods deteriorates when the model for deriving the data becomes more complex. Furthermore, this study revealed that likelihood methods were preferable when the data being analysed was based on a complex evolutionary model (Yang, 1996).

Bayesian inference methods have also been offered as alternative methods for the

construction of trees for phylogenetic inference, for example, MrBayes (Ronquist et al., 2012) and BEAST (Drummond and Rambaut, 2007). These methods allow for the use of various substitution models including user defined models, however, the main challenge with Bayesian methods is that they are computationally expensive using such complex calculations as the Markov Chain Monte Carlo (Yang and Rannala, 2012).

Although some of the approaches discussed here, such as Bayesian and maximum likelihood based methods have become popular their implementations are still poorly understood because of the complex algorithms on which they are based. On the other hand, distance based methods such as the UPGMA and NJ methods are still in use as they require less computational space and time as the algorithms compare fewer trees than character based methods such as parsimony and maximum likelihood (Yang and Rannala, 2012).

2.9 Models of sequence evolution

Models of sequence evolution have become a ubiquitous parameter in phylogenetic studies and numerous have been proposed (Delport et al., 2009). These models can be used to measure the selective pressure on sequences by determining the evolutionary rate at codon sites (Pond and Muse, 2005; Yang, 2007). They have also been used in determining those sites experiencing positive selection or distinguishing between coding and non-coding regions (Sadri et al., 2011). These models of sequence evolution are usually generated using a continuous time Markov process which attempts to describe mutations observed in the phylogenetic tree. While various models have been proposed, they vary with the number of parameters and complexity: the HKY85 by Hasegawa et al. (1985) and the Generalised time-reversible (GTR) by Tavaré (1986) being most widely used to model sequence evolution.

The earliest model of sequence evolution was proposed by Jukes et al. (1969) with a

single parameter for nucleotide substitutions. Thereafter, Kimura (1980) provided his model, called the K80, where he introduced a parameter to account for the transition and transversion ratio. Felsenstein (1981) also proposed the F81 model to account for unequal nucleotide frequencies and later another model, the F84 model, which accounts for differences in transition and transversion ratios and unequal nucleotide frequencies. The F84 model has also been found to bear some similarity to the HKY85 model.

Other models have since been proposed to account for bias in G+C content (Tamura, 1992), accounting for the different types of transition (Tamura and Nei, 1993). Studies have also proposed improved models, for example, Yang94 (Goldman and Yang, 1994) and MG (Muse and Gaut, 1994) which have become widely used. The Likelihood Ratio Test (LRT) (Anisimova et al., 2001) can be used to compare two hypotheses if the models are nested, for example, JC69 is the J81 model with equal base frequencies. The LRT involves comparing the log-likelihood values of the models in a chi-squared table. Conversely, the Akaike Information Criterion (AIC) (Posada and Buckley, 2004) can be utilised if the models are not nested. In this study models of sequence evolution will be used as parameters to calculate ω value profiles at amino acid sites in a protein sequence. These profiles will subsequently be used to compile the evolutionary rates profiles database which will be used to infer protein domain functions using an evolutionary rate based algorithm.

2.10 Phylogenetic analysis and rates of evolution

Various approaches of calculating rates of evolution using stochastic models have been widely utilised in phylogenetic and molecular studies (Delport et al., 2009). Phylogenetic analysis can also be done using Parsimony and Likelihood Methods (Yang, 1996) and guidelines for the application in phylogenetic analysis have been provided (Yang and Rannala, 2012). PAML is a sophisticated suite of software for phylogenetic analysis of DNA and protein sequences using a maximum likelihood (ML) approach (Yang,

2007). The PAML package also includes the CODEML program which was utilised by Durand et al. (2010) to determine the ω maximum likelihood estimates (MLE) at codon sites as input for the FIRE algorithm.

The CODEML program found on the PAML suite of programs has become the focus of numerous studies where it is used among other things for the determination of the MLEs at codon sites using models of sequence evolution. Furthermore, the program offers numerous models of sequence evolution. These models are Markovian model based and the canonical model found in the CODEML program is that of Goldman and Yang (1994) and the rate matrix of the instantaneous substitution of codon i for codon j taken from Yang (1998) is given by:

$$q_{ij} = \begin{cases} 0, & \text{if the two codons differ at more than one position,} \\ \pi_j & \text{for the synonymous transversion,} \\ \kappa\pi_j, & \text{for nonsynonymous transition,} \\ \omega\pi_j, & \text{for nonsynonymous transversion,} \\ \omega\kappa\pi_j & \text{for nonsynonymous transition,} \end{cases} \tag{2.5}$$

where κ represents the transition/transversion rate ratio, ω is the evolutionary rate (d_N/d_S), π_j represents the equilibrium frequency of codon j calculated from the data.

Numerous parameters can be specified on the CODEML program among them is the branch model parameter which allows for the specification of the heterogeneity of the ω ratio in the phylogenetic analysis; this parameter is important in detecting sites with positive selection. Several ω ratios can be specified using parameters related to the branch labels. While the CODEML program provides numerous models, this review will focus only on the M2a and M3 models. These models, M2a and M3, were used in the FIRE study to allow for the detection of positive selection at codon sites, these models will also be used in this study. Table 2.1 shows the parameters that are found in the M2a and M3 site models.

Table 2.1: The M2a and M3 site models and their parameters.

Model	NSsites	*No. of parameter	†Parameters
M2a (Selection)	2	4	$p_0, p_1(p_2 = 1 - p_0 - p_1)$, $\omega_0 < 1, \omega = 1, \omega_2 > 1$
M3 (Discrete)	3	5	$p_0, p_1(p_2 = 1 - p_0 - p_1)$, $\omega_0, \omega_1, \omega_2$

*No. of parameters specifies the number of parameters. †Parameters can be free or not free, those that are in parentheses should not be counted, therefore there are 4 and 5 parameters for the M2A and M3 site models respectively.

The number of parameters used for the analysis is crucial for Likelihood Ratio Tests (LRT) in the comparison of nested models, this number is then used as the degrees of freedom (Yang, 1998). Bayesian based methods such as the Naive Empirical Bayes (NEB) and the Bayes Empirical Bayes (BEB) have been provided to detect those sites under positive selection based on statistically significant likelihood ratio tests (Nielsen and Yang, 1998; Yang et al., 2005). The most common use of these Bayesian based methods is determining the site classes based on the computation of posterior probabilities. HyPhy (Hypothesis Testing using Phylogenies) is a powerful tool which uses statistical models to estimate selection pressures on alignments of coding sequences (Pond and Muse, 2005). Recent studies have proved HyPhy to be a formidable tool in detecting purifying selection in HIV-1 sequences (Ngandu et al., 2008).

In a study describing the Rate4Site algorithm, Pupko et al. (2002) identified functionally important regions of amino acid sequences using the evolutionary rate at codon sites. They used the ML approach was used to determine the evolutionary rate using phylogenetic trees generated by the UPGMA method. Pupko et al. (2002) used a modified PAML approach in modeling sequence evolution to determine the MLEs of the rates of evolution at amino acids sites. They conceptualised an approach of determining the function of proteins with known 3D structure by mapping the evolutionary rates shared by homologous proteins. Their work was based on the assumption that surface protein are constrained due to their binding nature. Finally, the Rate4Site program

infers functionally important regions of proteins using 3D structures combined with statistical information. Using this information the group was able to detect the SH2 peptide-binding groove domains.

2.11 The evolutionary rate

Zuckerkandl and Pauling (1965) proposed using the comparison of homologous sequences as a measure of selective pressure at coding sites. These comparisons reveal that substitutions at amino acid coding sites due to the degeneracy of the genetic code, fall into two classes of nucleotide substitutions: synonymous and non-synonymous substitutions. Synonymous substitutions do not result in a change in the amino acid coded for, while non-synonymous substitutions result in changes. The ratio of the rate of non-synonymous (d_N *or* K_a) to the rate of synonymous (d_S *or* K_s) substitutions is known as the evolutionary rate represented by the d_N/d_S or ω parameter proposed by Miyata and Yasunaga (1980). ω is used as an indicator of selective pressure acting on coding and non-coding regions (Kimura, 1984). Consequently, ω is used to detect the effects of natural selection on genes and as an indicator of molecular evolution. It measures the relative strength of evolutionary forces that have moulded a particular gene resulting in the protein under its control.

Comparative sequencing projects are carried out on one or more species to investigate and to identify if evolution has occurred in a particular gene (Jørgensen et al., 2005). Such projects provide an insight into actions of natural selection at the sequence level. In a paper on gene prediction, the ω ratio was used to identify protein coding regions between mouse and human exons (Nekrutenko et al., 2002). An ω greater than unity ($\omega > 1$) indicates that the rate of non-synonymous substitutions is greater than the rate of synonymous substitutions and this is indicative of positive selection. Positive selection also known as Darwinian selection provides evidence that mutations are conferring some advantageous fitness trait. Conversely, if the non-synonymous substi-

tution rate is less than the synonymous rate ($\omega < 1$), this is indicative of negative or purifying selection acting against these changes resulting in deleterious mutations. On the other hand, sites under purifying selection are under evolutionary constraints and are not likely to change and this is evidence of structural or functional importance. Sadri et al. (2011) and Jørgensen et al. (2005) used the guidelines discussed above to identify coding and non-coding regions. Additionally, those sites that are experiencing neutrality, when the non-synonymous and synonymous substitution rates are relatively equal ($\omega = 1$).

The challenge of estimating the ω has resulted in several proposed methods that range from approximation methods (Nei and Gojobori, 1986) to sophisticated maximum likelihood methods (Yang, 1996) based on probabilistic models of sequence evolution (Delport et al., 2009). Several programs for studying ω at codon sites or the selective pressure on amino acids are available such as ConSurf (Celniker et al., 2013), BADASP (Edwards and Shields, 2005) and CRASP (Afonnikov and Kolchanov, 2004). Online servers providing phylogenetic tools have also been provided, for example, Datamonkey for detecting sites under positive selection (Delport et al., 2010) and SWAKK, an online server that allows for the detection of those sites under positive selection based on a sliding window substitution rate approach (Liang et al., 2006).

Huelsenbeck et al. (2006) described a method for modelling the variation of the non-synonymous substitution rates on sequences using the Dirichlet process mixture model. This is similar to the method described by Yang and Nielsen (2000) where a Bayesian framework is used to estimate the parameters of a model. In a study developing the Rate4site algorithm focused on aligning SH2 domain surface protein, Pupko et al. (2002) found that their method was superior to other widely used methods. They attributed the success of their algorithm to the fact that the algorithm incorporates branch lengths in the evolutionary model.

The limitations of single-site analytic methods to detect positive selection were identified in a perspective paper by Antunes and Ramos (2007), where they suggest that single-site methods for detecting adaptive evolution had two main flaws: the increased probability of false positives and high values of d_N/d_S not due to adaptive evolution events but due to demographic events. In this study they provide computational approaches to detecting positive selection.

2.12 The efficacy of evolutionary rates as an alignment metric

Evolutionary rates are commonly used to identify genetic features of potential biological interest. However, no framework existed for comparing the evolutionary rate distribution between genes and using them for sequence alignment until the work done by Durand et al. (2010) and Pond et al. (2010). Durand et al. (2010) hypothesised that the selective pressures acting at codon sites across coding sequences, and not residue positional similarity, could be used to perform alignments. They tested the hypothesis that an evolutionary rate approach could be used to infer protein domain function in the absence of significant sequence similarity and the FIRE algorithm was a product of this work. Their results revealed that the FIRE algorithm in terms of quality of alignments could perform as well as conventional algorithms such as FATCAT, T-COFFEE, MAFFT and CLUSTAL (Durand et al., 2010). However, there were numerous false positives. These false positives were identified when two unrelated highly conserved domains of similar lengths were aligned for data set where the MLEs were less than 0.3, indicating the poor specificity of the algorithm.

A study based on a slightly similar approach by Clark and Aquadro (2009) described a novel method for detecting and identifying functionally co-evolving proteins, in their work they hypothesize that proteins that are functionally linked due to protein-protein interactions or proteins sharing common functions co-evolve are under similar

evolutionary pressures. This alternative approach is based on evidence that proteins that interact physically can influence the other's rates of evolution as their functions co-evolve. In their a paper using the network of Drosophilia Nuclear pore as an example of interacting proteins, they provided evidence that correlated evolution methods could be used to demonstrate functional relatedness of proteins. One of the challenges identified by Clark and Aquadro (2009) was the difficulty of making conclusions about correlated genes when they are of unequal length.

The findings of the FIRE study complemented the findings of Pond et al. (2010) where they demonstrated that the probability distribution of the evolutionary rate which they termed "Evolutionary Fingerprints" could be used as gene identifiers. Therefore, the study on FIRE algorithm provided a proof of concept in its novel approach to alignment as a formidable tool in sequence alignment. However, more work is required to improve its specificity and produce a robust tool for aligning sequences.

2.13 Biological databases

The large amounts of data generated by high throughput sequencing projects such as the Human Genome Project (Lander et al., 2001), the 1000 Genomes Project (Abecasis et al., 2010) has resulted in the increase in size of biological databases such as GenBank (Benson et al., 2010), PFAM (Bateman, 2004) and EMBL-Bank (Stoesser et al., 2002). The most widely used structural database is the Protein Databank which was developed as a repository of three-dimensional and structural information for proteins (Bernstein et al., 1977). Margaret Dayhoff who is also known as the "father and mother" of bioinformatics, and others compiled one of the first biological databases known as the Atlas of Protein Sequence and Structure which later became known as the Protein Information Resource (PIR) database (Wu et al., 2003).

GenBank is an annotated nucleic acid database of genomes of 260 000 species freely

accessible and together with the DNA DataBank of Japan (DDBJ) (Kaminuma et al., 2011) and the European Molecular Laboratory (EMBL) (Stoesser et al., 2002) constitute the largest collection of nucleotide sequences (Benson et al., 2010). Sometimes secondary databases contain information from primary database such as GenBank presented according to some organisation or classification. Popular examples of secondary databases include SCOP (***S***tructural ***C***lassification ***O***f ***P***roteins) (Murzin et al., 1995), CATH (Orengo et al., 1997), PROSITE (Falquet et al., 2002) and eMOTIF (Huang and Brutlag, 2001). Composite databases are an amalgamation of primary databases. The OMIM (Online Mendelian Inheritance in Man) is a composite database of proteins and their associated genetic diseases (Hamosh et al., 2005). The GenBank database retrieval system: Entrez, which is integrated with annotation data such as protein structure and domain information also allows for access to literature through the PubMed database. The BLAST algorithm is also used for sequence similarity searches on GenBank and related databases (Altschul et al., 1990).

The Universal Protein Knowledge Base (UNIPROTKB) comprises two databases: the high quality and manually curated and reviewed, SWISS-PROT and the automatically generated and annotated TrEMBL database. The CATH (Orengo et al., 1997) database uses a Needleman-Wunsch based algorithm for sequence comparison and the SSAP (***S***equential ***S***tructure ***A***lignment ***P***rogram) for protein structure comparison algorithm (Taylor and Orengo, 1989) for structural comparison to cluster structures from the Protein Data Bank (Bernstein et al., 1977). The rationale behind the CATH database is that a structure based approach is more effective in detecting functional or structural relationships between proteins especially in cases where sequence identities are low. Proteins in the CATH database are organised in a hierarchical order with five major levels; class (C), architecture (A), Architecture(A), Topology (T), Homologous superfamily (H) and sequence family. High throughput projects have resulted in increased rates at which sequences are identified has created a demand for fast classifi-

cation and predicting methods. Consequently, a semi-automatic approach was chosen for CATH to keep up with the phenomenal rates at which sequences are identified. Murzin and colleagues developed the SCOP database where proteins are grouped into homologous groups using sequence similarity (Murzin et al., 1995). In the database, classification of distantly related structures is done manually, evolutionary relationship and literature are also used to classify protein on the SCOP database. On the other hand, the DALI method (Holm et al., 2008) is an approach that uses a weighted sum of families of intra-molecular distances to cluster members of the PDB database using structures of proteins to produce the FSSP database (Holm and Sander, 1994).

2.13.1 The PFAM database

One of the objectives of this study is to compile a database of evolutionary rate profiles of the PFAM database. These evolutionary rate profiles will be queried using a new implementation of the algorithm to infer domain functions. In the PFAM database (Bateman, 2004) proteins are classified into domains of shared homology. The domains of functional regions are used to construct MSAs of each family from proteins with shared homology. The HMMER3 algorithm based on a Hidden Markov Model (HMM) approach is used to detect family members with a significant similarity (Mistry et al., 2013). The PFAM database consists of two separate databases: PFAM-A consisting of manually curated families and PFAM-B which is automatically generated and of lower quality. HMM profiles of each family are generated using seed alignments; these are then used to find other members of the family using HMMER3 (Mistry et al., 2013) algorithm from the UNIPROTKB repository (Consortium et al., 2013). The PFAM-A database (release 27.0) contains 13 672 families, the database also has clans which are high quality families of PFAM-A generated from alignments of high levels of similarity. Phylogenetic trees of each PFAM family are also available for download from the PFAM database these are calculated using the FASTREE program using the Neighbour-Joining algorithm (Price et al., 2010).

The increase in high throughput data projects such as the 1000 genomes project (Abecasis et al., 2010) and the resultant increase in primary database raises challenges for the future of databases; new approaches in the areas of organising the database and retrieval of information are required. Other challenges that arise with the growth of databases are increases in mis-annotations and the resultant decrease in reliability of databases size arise as the annotation is automated. PFAM faces these challenges and has resolved to offer two databases as mentioned before: the reliable and manually curated PFAM-A and the computer curated PFAM-B. Another solution which has been suggested to prevent mis-annotations is to identify those annotations that have not yet been proven experimentally. It is important to highlight the importance of the searching algorithm as databases are only as good as their retrieval system for example BLAST and GenBank.

2.14 The "twilight zone" of sequence alignment

Structural comparisons have always been the standard of truth in the inference of phylogenetic relationships of distantly related proteins even where sequence algorithms have failed to detect phylogenetic relationships (Brenner et al., 1998). At sequence similarities lower than 25% it is difficult to infer homology due to sequence divergence over time. Doolittle (1981) coined the term "twilight zone" to describe those alignments that are within the 20% to 30% region of sequence alignment (Rost, 1999). Thompson and others compared the performance of algorithms, in this study they identified the importance of improving the performance of algorithms in those regions below the twilight zone (Thompson et al., 1999).

The inference of structural, functional or evolutionary relationships of novel proteins is a common problem in molecular biology. It is general practice to determine the structure of a novel protein by searching against some structural database such as the PDB (Bernstein et al., 1977) or functional domain database such as PFAM (Bate-

man, 2004) and once matches are found inferences can then be made. The domains of proteins become shuffled during evolution (Bork, 1991); however, the structural cores of these proteins remain unchanged and these similarities can be used to infer evolutionary relationships in different species (Shoemaker et al., 2006; Orengo et al., 1997). In addition, functional domains of protein sequences with less than 30% similarity have been found to share similar fold configurations and these can be detected even in sequences that have undergone extensive divergence (Chothia and Lesk, 1986; Orengo et al., 1997). The information provided by the 3D structure of proteins, can be used to infer evolutionary or structural relationships even in distantly related proteins. Structural or functional similarities shared by protein domains with very low similarity are suggestive of a common evolutionary origin and this was the basis of classifying proteins used in CATH (Orengo et al., 1997).

It has also been noted that conventional sequence alignment approaches fail to detect the homology in rapidly evolving, evolutionary distant species or sequences that have evolutionary bias. Studies have demonstrated that sequences with identities lower than the "twilight zone" can share similar structures and function (Orengo et al., 1997). Sequences with less than 15% identity have been identified in many protein pairs which share similar structures and functions for example in the globin family (Chothia and Lesk, 1986; Orengo et al., 1994). Therefore, structural data can provide insight into the relationships between proteins even when the proteins have low sequence identities (Orengo et al., 1999). On the other hand, the determination of some protein structures is challenging and alternative approaches for inferring functions when sequence similarity is low are required.

2.15 The enigmatic HBx protein

The Hepatitis B Virus (HBV) has been implicated in diseases such as Hepatocellular carcinoma (HCC) (Liang, 2009), chronic hepatitis and liver cirrhosis affecting millions

of people in the world. Some studies suggest that the virus could increase the development of pancreatic cancer. Understanding the replication and pathogenesis of the virus would assist efforts in eliminating the virus and save lives. One of the challenges in understanding the biology of the HBV has been the failure to elucidate the numerous functions of the Hepatitis B Virus X protein (HBx) protein which is a protein produced by HBV involved in the virus replication (Colgrove et al., 1989). The viral DNA identified in patient's DNA suffering from HCC has been identified to be HBx. This study focuses on the X protein because its function is not well understood (Bouchard and Schneider, 2004).

A member of the Hepadnaviridae family, the HBV has only been identified to affect human hosts, however, variants of the virus in the same family have been identified in apes, woodchucks, squirrels and avians such as ducks, geese and cranes (Chang et al., 2001). The X protein has only been identified in the virus affecting mammals which may suggest the virus is only necessary for mammals. It is believed that this could be linked to a response to the strength of the mammalian immune defence mechanisms (Chang et al., 2001). It has been found to be transmitted through the exchange of fluids, however, the method of transmission of the virus is still not yet fully understood. DNA analysis has revealed that it possesses a circular partially double stranded DNA and the viral genome has been found to code for 4 known genes C, X, P and S (Ganem and Varmus, 1987). Once the virus has entered a host, it infects the hepatocyte cells where it transports its partially double stranded DNA to the nucleus where it is converted to covalently closed circular double stranded DNA (cccDNA) where it used for viral transcription using the host machinery.

The Hepatitis B X (HB x) protein is 154 amino acids in size with a molecular mass of 17.5 kDa. This protein was historically named the X protein because at the time of its discovery it had an unknown function, however, some functions of this enig-

matic protein have been established experimentally (Bouchard and Schneider, 2004). The lack of understanding of the HBx structure has prompted numerous investigations of its functions. Experiments have failed to conclusively identify the role played by the protein in the hepadnavirus life cycle in mammalians (Madden and Slagle, 2001). Consequently, there has been a lot of controversy and speculation about its structure and function. This controversy is attributed to lack of homology with any known protein in biological databases, exacerbated by the fact that the structure is not well understood (Murakami, 1999). Researchers have failed to determine its structure using high-resolution crystallization and Nuclear Magnetic Resonance (NMR) methods (Murakami, 1999).

Experimental evidence would also suggest that the protein interacts with the signalling pathways in the cytoplasm which are involved in up-regulating viral replication and cell proliferation (Madden and Slagle, 2001). Studies of the function of the HBx protein have also shown that the HBx protein is moderately involved in the transcription activation of promoters and enhancers (Yen, 1996). Some experiments also suggest that the X protein is involved in the trans-activation of viral and cellular genes where it is thought to interact with some transcription factors and co-activators (Colgrove et al., 1989). The protein also interacts with nuclear transcription components and activates cytosolic signal transduction pathways involved in such activities such as cell apoptosis (Madden and Slagle, 2001). The X protein has also been implicated in the regulation of viral replication where it facilitates viral replication (Xu et al., 2002).

On the other hand, the protein has also been found to co-act with TAp73 which favours cell survival, these and other process are cell dependent where it triggers apoptosis in transformed cells and preventing apoptosis in normal hepatocyte cells (Ganem and Varmus, 1987). Additionally, it has been found to prevent apoptosis by inhibiting the pro-aptotic P53 protein by direct interaction or by preventing its entry into the

nucleus (Terradillos et al., 1998). Furthermore, it has also been found to interfere with host cell defences where it counteracts the mammalian defences system which could result in oncogenesis (Liang, 2009). Experiments have also shown that the X protein promotes the degradation of mitochondrial antiviral-signaling protein (MAVS) a protein required for innate immune defence and that this can also result in cell apoptosis (Kumar et al., 2011). These and other factors suggest that the virus is engaged in inflammation and hepatocellular carcinoma development.

In this study, a new implementation of the FIRE algorithm will be used to infer the functions of the HBx protein. The objective is to find a protein with an evolutionary rate or ω profile similar to that of the HBx protein. The rationale is that a protein with a similar evolutionary rate profile will share similar domain function as HBx. The proof of concept for such an approach has already been provided by Durand et al. (2010) and Pond et al. (2010). Therefore, since the FIRE algorithm proved effective in alignments of low sequence similarity, using an algorithm based on a similar hypothesis will be a fresh approach in inferring domain functions of HBx. Using such a novel approach could provide some insights into the domain functions of this enigmatic protein. The results of such a study may help in understanding the role of HBx in virus replication, clear some of the controversy around its function and ultimately lead to the elimination of the HBV.

Chapter 3

Methods

In this chapter the implementation and evaluation of the BLOSUM-FIRE algorithm is described. The old algorithm will be referred to as FIRE and the new algorithm developed and implemented in this research will be referred to as BLOSUM-FIRE. The chapter begins by describing the formulation of the current algorithm and its evaluation using real and simulated data. The chapter then describes the approaches used to conceptualise and implement the new algorithm. A method for empirically determining the gap penalties based on a variable gap function in the new algorithm is presented. An approach to evaluate BLOSUM-FIRE is also presented which also describes a framework for comparing the performance of the new algorithm with widely used sequence alignment algorithms. The chapter also provides a description of the compilation of the Evolutionary rates Database (EvoDB) using the PFAM, SWISS-PROT, TrEMBL and GenBank databases. The chapter is concluded by describing an approach to infer the domain functions of the enigmatic and controversial Hepatitis B Virus X protein whose functions have been challenging to infer.

3.1 The FIRE algorithm

The FIRE algorithm is a command line tool implemented in the Python programming language (Van Rossum and Drake Jr, 1995). The algorithm finds the optimal global alignment of two protein sequences using the maximum likelihood estimates (MLEs) of

the evolutionary rate ($\omega = d_N/d_S$) at codon sites. It requires two MSAs of nucleotide sequences with their corresponding phylogenetic tree files to generate a pairwise amino acid alignment as illustrated in Figure 3.1. A modified Needleman-Wunsch algorithm determines the optimal alignment (Section 2.1.1). MSA files and their corresponding phylogenetic tree files are used as input for the CODEML program (Section 2.10 and Section 3.2 for methods used) to produce the MLEs of ω at codon site. The CODEML rst output files are then used as input for the algorithm to generate the final alignments.

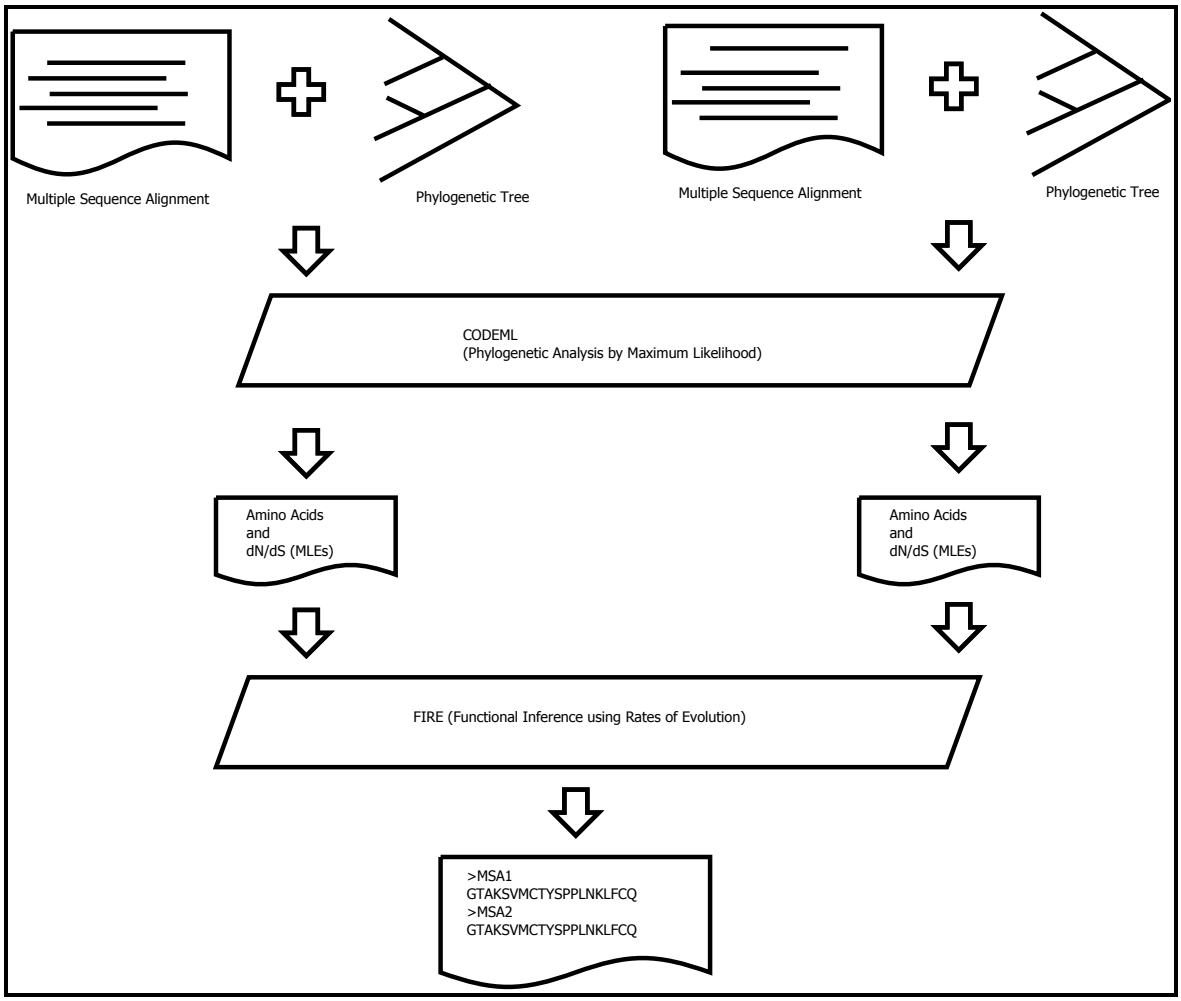


Figure 3.1: Generating alignments using the FIRE algorithm. The work flow for generating alignments from the nucleotide MSAs and corresponding phylogenetic trees to generate the aligned protein sequences in FASTA format is described.

The cost function of the algorithm uses a codon score (cs) as a scoring metric for the similarity between two aligned ω MLE values obtained from an rst CODEML output

file. The FIRE algorithm does not use the residue identity of the amino acids, instead the algorithm generates alignments based solely on maximising the similarity of the ω MLEs values at codon sites. Let ω_i represent the evolutionary rate at site i in sequence M and ω_j represent the evolutionary rate at site j in sequence N. To calculate the similarity score, the following equation is used:

$$cs(\omega_i, \omega_j) = 1 - \frac{|\omega_i - \omega_j|}{\omega_{max}} \tag{3.1}$$

where ω_{max} is the maximum difference (capped to 1.5), if this is exceeded then Equation (3.2) is used, such that:

$$cs(\omega_i, \omega_j) = 0, \quad \text{if} \quad |\omega_i - \omega_j| > \omega_{max} \tag{3.2}$$

The rationale behind capping ω_{max} at 1.5 was to emphasize those sites under positive selection rather than the absolute difference between ω_i and ω_j. The cs for two aligned sites was normalised to the range [0,1]. The FIRE score of an alignment is the sum of the cs at aligned codon sites normalized by the length of the longest sequence. An affine gap penalty (Section 2.5) model was implemented for the initiation of gaps and extension of the gaps as a function of the length of the gap. The affine gap penalty is calculated by:

$$P = g + el \tag{3.3}$$

where P is the affine gap penalty total, g is the gap open penalty for gap initiation, e is the gap extension penalty and l is the length of the gap. Empirically determined values of 0.5 and 0.1 are set as the default parameters for the gap open and extension penalties respectively. The algorithm produces alignments in conventional amino acid sequence alignment formats with FASTA set as the default. Alignments in all formats are generated using functions imported from the Biopython Module (Cock et al., 2009). Percentage similarity plots and histograms can be obtained for alignments as optional parameters passed to the algorithm.

3.2 Calculating ω MLEs profiles

The CODEML program, found on PAML version 4.4 (Yang, 2007), was used to determine the posterior probabilities of Bayes Empirical Bayes (BEB) of the MLEs of ω values at codon sites. The program provides parameters for selecting models of sequence evolution, models of sequence evolution are discussed in Section 2.9. The M2a (Positive selection) model parameter which detects sites under positive selection was chosen for all analysis. This model was chosen to identify those sites under positive selection. The CODEML options: $F3 \times 4$ codon model, $Model = 0$ and $NSsites = 2$ were used to generate the MLEs of the ω. The model and other parameters for the CODEML program were specified in a *codeml.ctl* file provided in Figure 3.3. This file was read during the execution of the CODEML program which generated several output files. It contains the BEB of MLEs for ω at codon sites for the MSA.

An example of a truncated CODEML *rst* is presented in Figure 3.2. The *rst* files were then used as input for the FIRE to generate alignments as shown in Figure 3.1. There are several columns in the file, including the BEB probabilities of the 3 classes of ω values: $\omega>0$, $\omega = 1$ and $\omega<1$ respectively located after the column of the amino acids residues. The ω MLEs at codon sites values were found in the 7th column of the *rst* file and these were used as an indicator of selective pressure and used by the algorithm to generate alignments.

```
   1 V   0.71293 0.26932 0.01775 ( 1)  0.438 +-  0.437
   2 S   0.77526 0.21165 0.01308 ( 1)  0.382 +-  0.404
   3 G   0.09223 0.78457 0.12320 ( 2)  1.082 +-  0.702
   4 S   0.95671 0.04162 0.00167 ( 1)  0.219 +-  0.192
   5 P   0.94112 0.05656 0.00233 ( 1)  0.232 +-  0.219
   6 G   0.94136 0.05618 0.00246 ( 1)  0.232 +-  0.221
   7 K   0.39896 0.55329 0.04775 ( 2)  0.728 +-  0.536
   8 T   0.93390 0.06281 0.00329 ( 1)  0.239 +-  0.237
   9 I   0.30155 0.61834 0.08011 ( 2)  0.866 +-  0.715
  10 T   0.63753 0.33116 0.03131 ( 1)  0.518 +-  0.531
```

Figure 3.2: A CODEML *rst* file used as input for the algorithm, showing the position of amino acids and the posterior probabilities of the 3 classes of ω. The ω values are used by the algorithm to generate alignments are located after the column with brackets.

```
      seqfile = kappa.nuc * sequence data filename
     treefile = kappa.trees      * tree structure file name
      outfile = test1.txt           * main result file name
        noisy = 0  * 0,1,2,3,9: how much rubbish on the screen
      verbose = 1  * 0: concise; 1: detailed, 2: too much
      runmode = 0  * 0: user tree;  1: semi-automatic;  2: automatic
                   * 3: StepwiseAddition; (4,5):PerturbationNNI; -2: pairwise
      seqtype = 1  * 1:codons; 2:AAs; 3:codons-->AAs
    CodonFreq = 2  * 0:1/61 each, 1:F1X4, 2:F3X4, 3:codon table
       *ndata = 5
        clock = 0  * 0:no clock, 1:clock; 2:local clock; 3:CombinedAnalysis
       aaDist = 0  * 0:equal, +:geometric; -:linear, 1-6:G1974,Miyata,c,p,v,a
   aaRatefile = wag.dat  * only used for aa seqs with model=empirical(_F)
                   * dayhoff.dat, jones.dat, wag.dat, mtmam.dat, or your own
        model = 0
                   * models for codons:
                       * 0:one, 1:b, 2:2 or more dN/dS ratios for branches
                   * models for AAs or codon-translated AAs:
                       * 0:poisson, 1:proportional, 2:Empirical, 3:Empirical+F
                       * 6:FromCodon, 7:AAClasses, 8:REVaa_0, 9:REVaa(nr=189)
      NSsites = 2  * 0:one w;1:neutral;2:selection; 3:discrete;4:freqs;
                   * 5:gamma;6:2gamma;7:beta;8:beta&w;9:beta&gamma;
                   * 10:beta&gamma+1; 11:beta&normal>1; 12:0&2normal>1;
                   * 13:3normal>0
        icode = 0  * 0:universal code; 1:mammalian mt; 2-10:see below
        Mgene = 0
                   * codon: 0:rates, 1:separate; 2:diff pi, 3:diff kapa, 4:all diff
                   * AA: 0:rates, 1:separate
    fix_kappa = 0  * 1: kappa fixed, 0: kappa to be estimated
        kappa = 2  * initial or fixed kappa
    fix_omega = 0  * 1: omega or omega_1 fixed, 0: estimate
        omega = .4 * initial or fixed omega, for codons or codon-based AAs
    fix_alpha = 1  * 0: estimate gamma shape parameter; 1: fix it at alpha
        alpha = 0. * initial or fixed alpha, 0:infinity (constant rate)
       Malpha = 0  * different alphas for genes
        ncatG = 10  * # of categories in dG of NSsites models
        getSE = 0  * 0: don't want them, 1: want S.E.s of estimates
 RateAncestor = 1  * (0,1,2): rates (alpha>0) or ancestral states (1 or 2)
   Small_Diff = .45e-6
    cleandata = 1  * remove sites with ambiguity data (1:yes, 0:no)?
  fix_blength = 0  * 0: ignore, -1: random, 1: initial, 2: fixed
       method = 0  * Optimization method 0: simultaneous; 1: one branch a time
* Genetic codes: 0:universal, 1:mammalian mt., 2:yeast mt., 3:mold mt.,
* 4: invertebrate mt., 5: ciliate nuclear, 6: echinoderm mt.,
* 7: euplotid mt., 8: alternative yeast nu. 9: ascidian mt.,
* 10: blepharisma nu.
* These codes correspond to transl_table 1 to 11 of GENEBANK.
```

Figure 3.3: The *codeml.ctl* file showing the parameters used by the CODEML executable. Parameters with the asterisk (*) are not required for the model used; the program does not read any information after the asterisk, these are used for comments in the file. The parameters: *model* and *NSsites* were used to specify the models of nucleotide substitution and site models respectively and of particular importance for determining the ω MLEs used in this study.

3.3 Evaluation of the FIRE algorithm using real data

The FIRE algorithm was evaluated using data sets identified in the original concept paper of Durand et al. (2010). The BEB of ω MLEs for data from the initial FIRE study were collected and experiments were carried out to evaluate the algorithm. The data included the following domains: a highly conserved transcription factor MYB1 DNA-binding domain (DBD) orthologs of protozoa and metazoa, a conserved tumor suppressor p53 DBD, a metabolic enzyme glycerol kinase (GK), κ light chain antibody and λ light chain antibody variable regions. This initial stage of the study provided insight into the scoring function of the algorithm. Those data sets that produced false positives were identified, analysed and the correlation between the statistical distribution of the ω values and the quality of alignment was also investigated.

To further investigate the occurrence of false positives, 100 alignments of evolutionary rate profiles of the families from the PFAM-A database were utilised. The evolutionary rate profiles were obtained from determining the MLEs at codon sites for 100 PFAM families using the CODEML program. These PFAM families share low sequence similarities. The alignments were generated using the FIRE alignment and the number of residues aligned were counted as a measure of quality of alignment. The same alignments were then generated using the MAFFT version 7.130b and CLUSTAL Omega version 1.2.0 algorithms. Each alignment was scored using the numbers of aligned residues, normalised by the maximum sequence length.

3.3.1 Evaluation of the algorithm using simulated data sets

The evaluation of FIRE using real data in the previous Section 3.3, revealed that the algorithm had reduced specificity in highly conserved data sets resulting in numerous false positives. Simulated data sets were created to further investigate scenarios result-

ing in false positives. Highly conserved datasets were simulated such that the ω values had a variance in the range $[0, 0.02]$ across all sites. Alignments were generated using FIRE default parameters for gap open and gap extension penalties. The simulated datasets were then aligned using the CLUSTAL Omega algorithm (Sievers et al., 2014) and MAFFT algorithm using default parameters. The quality of alignments from the two approaches were analysed and compared.

3.4 Conceptualizing a new approach for FIRE

Analysis of datasets and alignments in the experiments described in Section 3.3 and 3.3.1 provided a framework for the conceptualisation of a new algorithm. Initial evaluation of the algorithm revealed that the evolutionary rate on its own was not specific enough as a parameter for the alignment metric. In cases where the variance of ω was low, the approach had reduced specificity. A new scoring function was required to incorporate the identity of the amino acids using a substitution matrix, such as BLOSUM, PAM or Gonnet matrices (Section 2.4). It was also realised that information regarding the conservation of amino acids was untapped in FIRE. An objective to further explore information presented by the ω MLEs and include amino acid conservation data in the new algorithm was adopted.

The BLOSUM matrix and the PAM matrix have become the gold standards of substitution matrices however, the BLOSUM matrix (Section 2.4.2) has generally been found to be superior (Henikoff and Henikoff, 1993). The BLOSUM matrix has become the *de facto* substitution matrix and can be found in popular programs such as BLAST (Altschul et al., 1990) and CLUSTAL W (Thompson et al., 1994). The BLOSUM substitution matrix was introduced to increase the sensitivity of the BLOSUM-FIRE algorithm especially in highly conserved datasets. A new BLOSUM-FIRE approach was proposed that used both the BLOSUM principle and FIRE principle. The following factors were considered in implementing the new scoring function:

- The selective pressure on the individual amino acids inferred from ω MLEs.
- The similarity of the ω MLE values (The FIRE principle).
- The BLOSUM score, $s(i, j)$ of two aligned amino acid residues (The BLOSUM principle).
- Scaling the BLOSUM score according to the selective pressure on the amino acids.
- Scoring the similarity of ω MLEs at codon sites using scores comparable to the BLOSUM matrix scores.

3.4.1 Comparison of a FIRE and BLOSUM based approach

It was hypothesised that incorporating residue identity using the BLOSUM would make the algorithm more robust. In order to evaluate the efficacy of a BLOSUM and FIRE coupled approach, the two approaches were compared against each other to identify strengths and limitations in the two approaches. To avoid inconsistencies and systematic errors that could be introduced by using a different approach, the algorithm scoring function was modified to use BLOSUM matrix scores to generate alignments. Alignments were generated using the default parameters empirically determined by Durand et al. (2010); gap open and gap extension penalties of 0.5 and 0.1 respectively for the FIRE algorithm. Penalty values of 7 and 2 were used for the gap open and extension penalties for the BLOSUM62 based algorithm, these represent a consensus of penalties obtained from empirical studies (Reese and Pearson, 2002; Pearson, 1995). The two approaches: FIRE and BLOSUM were compared against each other on data sets:

1. Highly conserved functionally similar sequences: Light chain antibody κ and λ antibody light chain regions.
2. Highly conserved unrelated sequences: P53 DBD and metazoan MYB.
3. Unrelated sequences of varied conservation: P53 DBD and antibody κ antibody regions.

3.4.2 Incorporating a BLOSUM scoring matrix into the FIRE algorithm

The results of the comparison of the approaches using experiments described in Section 3.4.1, revealed how the two, BLOSUM and FIRE, could be coupled. As these approaches were based on different approaches, it was hypothesized that they would complement each other. To utilise the Needleman-Wunsch DP algorithm (Section 2.1.1) a single score from FIRE and BLOSUM was required. To address the challenge of coupling the FIRE approach and BLOSUM approach, the two approaches were compared. Table 3.1 shows the main principles of the two approaches that were coupled in the BLOSUM-FIRE algorithm. For two amino acids i and j with evolutionary rates of ω_i

Table 3.1: Comparison of FIRE and BLOSUM substitution matrix approaches to sequence alignment. These guidelines were considered when coupling the two approaches.

The BLOSUM matrix approach	**The FIRE approach**
Uses the amino acid residue identity.	Uses the ω at codon sites ignoring residue identity.
BLOSUM values are integers ranging from negative to positive values differing in magnitude.	A FIRE score is the similarity of omega values normalized between zero and one.
BLOSUM values are empirically determined from blocks of conserved aligned amino acids. Conservation is only considered at sequence level.	Evolutionary rate provides information about the conservation at the amino acid level, hence an indicator of the selective pressure at the amino acid sites.
BLOSUM scores are discontinuous values ranging from negative numbers indicating strong mismatches to positive numbers indicating strong matches at amino acid level.	The FIRE score is a continuous value normalised between 0 and 1, with 1 indicating strong functional relatedness and 0 indicating little or no evidence of shared functions.
The BLOSUM score is an indicator of residue identity and does not account for the selective pressures at codon sites.	The FIRE score is an indicator of similarity of the selective pressure at codon sites.
Scores alignments for two amino acids sequences, a pairwise alignment.	Scores alignments of two amino acid sequences using ω at codon sites, a pairwise alignment.

and ω_j, the guiding principles for implementing the new algorithm included:

1. Determine the BLOSUM score $s(i, j)$, from the BLOSUM matrix using the identities of aligned residues.
2. Determine the selective pressure on the amino acids using the values of ω_i and ω_j.
3. Scale the BLOSUM score using the following principles of selective pressure: ω>1 indicates positive selection, ω <1 indicates negative selection or purifying selection and ω= 1 indicates neutral selection.
4. Determine the similarity of ω_i and ω_j values normalised between zero and one to a score called *similarity*.
5. Use the selective pressure to scale a BLOSUM score to a BLOSUM comparable score called the *selection*.
6. Add the *similarity* score to the *selection* score to obtain the BLOSUM-FIRE score.
7. Use a modified Needleman-Wunsch algorithm and new BLOSUM-FIRE scoring function to determine the mathematically optimal alignment.

3.5 Formulation of a novel scoring scheme

To combine the BLOSUM matrix based approach and FIRE approach, two scores were considered: (1) The selection adjusted BLOSUM score obtained from aligning the amino acid residues called *selection*; (2) The similarity of ω at codon sites scaled to BLOSUM comparable values called *similarity*. The main objective was to find a consensus scoring function for the two approaches so that one approach does not outweigh the other. The two approaches were independent of each other and the only overlap was the use of selective pressure to adjust BLOSUM score. To combine the two approaches a single value was required to score the *similarity* of the evolutionary rates and *selection* of selective pressure adjusted BLOSUM scores. The general equation for coupling the two was proposed to be in the form:

$$BLOSUM\text{-}FIRE = similarity + selection$$

where *similarity* and *selection* are determined using the guiding principles formulated in Section 3.4.2. Therefore, the proposed BLOSUM-FIRE score is the combined scores of the selection pressure adjusted BLOSUM matrix score and score the similarity of ω values for the aligned amino acids. An additive approach was identified as it fit into BLOSUM substitution matrix framework where the value of the BLOSUM score indicates physiochemical similarities of amino acids aligned (Section 2.4). The ω value of the aligned amino acids presents additional evidence on the conservation of the aligned amino acids, therefore the relationship between the two was proposed to be additive. However, the two scores are converted to comparable values for the addition to be achieved as BLOSUM values are are log-odd scores and ω is similarity score normalized in the range [0,1].

3.5.1 Approximating selective pressure and scaling the BLOSUM score

The ω values provide information on the level of conservation at amino acid level and such information can be used as an indicator of the functional role of the amino acid in the protein. A ω value close to zero indicates strong functional constraints, a ω >1 means non synonymous substitutions may offer some advantage to the function or structure of the protein concerned and high amino acid variability at such a site is expected. The evolutionary rate is discussed in Section 2.11.

The BLOSUM score is the log-likelihood score of the two amino acid residues being aligned. Section 2.4 provides the theoretical framework for substitution matrices. To conceptualise the equation for approximating selective pressure, a monotonic function was required. Studies have found that the selective pressure and conservation on a particular amino acid site decrease as the omega values increase (Section 2.11). Such a framework has also been used to identify coding regions of DNA from non-coding DNA (Nekrutenko et al., 2002; Jørgensen et al., 2005). Therefore, an equation was

chosen to model the correlation between selective pressure inferred from the ω values of the aligned amino acid sites. If a is the ω value of $amino\ acid_i$ for sequence M and b is the ω value of $amino\ acid_j$ from sequence N, then the proposed approximation for selective pressure on the two amino acids is calculated as follows:

$$sel_{ab} = e^{-(a+b)} \tag{3.4}$$

where sel_{ab} is the approximated selective pressure at the codon site in the range [0,1]. The selective pressure (sel_{ab}) provides information that is untapped by conventional alignment algorithms. Such an approximation given by Equation (3.4) emphasises the quality of alignments based on the level of conservation at the amino acid level. The rationale behind adding the ω values for the two amino acids is to obtain a measure of the selective pressure at that aligned site, using this framework the approximated selective pressure at the aligned site can be calculated using Equation (3.4). Once the selective pressure on the amino acid is obtained the BLOSUM score is scaled and adjusted according to the selective pressure sel_{ab} to obtain a BLOSUM scaled score called *selection* using the following equation:

$$selection = BLOSUM - |BLOSUM|(1 - sel_{ab}) \tag{3.5}$$

Equation (3.5) shows how the BLOSUM score was scaled by adjusting the absolute value. However, this equation was adjusted as it penalised negative scores. Equation (3.5) was then adjusted and this resulted in the simplified form:

$$selection = BLOSUM \times sel_{ab} \tag{3.6}$$

The rationale was that negative score already indicates a mismatch; using the form in Equation (3.5) penalises mismatches more and this can create unnecessary constraints in the alignments produced.

3.5.2 The FIRE approach

The premise of the FIRE principle is that amino acids under similar selective pressures share similar functions. It was therefore hypothesised that the spatial distribution of selective pressures along the amino acid sequence can be used to infer functional, structural relationships and possibly provide evidence of common ancestry. Results had shown that FIRE was able to align sequences while ignoring the residue identity of the amino acids. Further analysis proved that the algorithm produced false positives for certain data sets (MLEs of ω <0.3) (Durand et al., 2010). Therefore, a new approach was required to improve the specificity and sensitivity of the algorithm whilst preserving the principle it was based on. Let a be the evolutionary rate (ω) of amino acid i and b the ω of amino acid j, where i and j are indexes in amino acid sequences, then the similarity sim_{ab} of the evolutionary rates can be calculated such that:

$$sim_{ab} = 1 - \frac{|a-b|}{max(a,b)} \quad \text{if } a \text{ or } b \geq 1 \tag{3.7}$$

$$sim_{ab} = 1 - |a-b| \quad \text{if } a \text{ and } b < 1 \tag{3.8}$$

Either Equation (3.7) or Equation (3.8) can be used to calculate sim_{ab} to the absolute difference normalised in the range [0,1]. These equations were proposed to replace the Equations (3.1) and (3.2) this was to make sim_{ab} monotonic. Therefore, the function maximises the absolute differences at codon sites.

3.5.3 Coupling the BLOSUM approach with the FIRE approach

The FIRE principle uses the evolutionary rate to detect shared functions whilst the BLOSUM principle uses the amino acid identity to detect similarities. The main objective of this work was to improve the sensitivity and specificity of FIRE whilst preserving the FIRE principle. Therefore, a scoring function was required that would allow the two approaches to be combined, given the discontinuous and independent nature of

the BLOSUM values it was decided to scale the *similarity* score (sim_{ab}) values to BLOSUM comparable values.

To score the two approaches simultaneously a proportionality constant was proposed to make the sim_{ab} value comparable to BLOSUM scores. The next challenge was how to score the similarity of the evolutionary rate and the residue matches and mismatches in such a way that both approaches are comparable to each other.

It was important that such an approach should be one that used a function that would produce scores that would simplify the scoring of the algorithm. An assumption that an amino acid match is analogous to similarity of evolutionary rates was adopted. Therefore, a match score in amino acid residues was proposed to be equivalent to a FIRE score of 1.

3.5.4 Determining the BLOSUM-FIRE proportionality constant (K)

The proportionality constant (K) was proposed for the evolutionary rate approach score or sim_{ab} to be comparable to BLOSUM scores. Let sim_{ab} from Equations (3.7) or (3.8) be the similarity between the ω values at sites a and b, therefore the new score is given by:

$$BLOSUM' = K \times sim_{ab} \tag{3.9}$$

where $BLOSUM'$ is a BLOSUM comparable score and K is the proportionality constant obtained empirically from the substitution matrix, for example, BLOSUM62. Equation (3.9) was based on the assumption that a BLOSUM identical amino acid match score was analogous to a sim_{ab} score of 1. The new BLOSUM-FIRE score was given by:

$$BLOSUM\text{-}FIRE = BLOSUM \times sel_{ab} + K \times sim_{ab} \tag{3.10}$$

The K was empirically determined using a Student's two sample mean t-test. Two CODEML output rst files were modified by inserting all the amino acids in the amino acid alphabet and the ω values at codon sites were set to 0.00, indicating the highest level of conservation and the highest selective pressure. These were then used as input for the FIRE algorithm. The algorithm was modified to obtain all the identical amino acid match scores and to iteratively change the K value and compare it with the mean of the match scores. Therefore, K was iteratively increased until it was within the 95% confidence level using the Student's two sample mean t-test and it was found that $K = 5.63$.

3.5.5 The BLOSUM-FIRE scoring function

The strength of the FIRE approach is its ability to align sequences using ω values at low sequence similarity without using the amino acid identity. The BLOSUM approach has been found to be effective at aligning sequences using amino acid identity. To increase the specificity of the algorithm the areas of strengths for the BLOSUM and FIRE approaches were incorporated into Equation (3.10), this was conceptualized using guidelines described in Section 3.4.1. The FIRE algorithm produced false positives when sequence conservation was high. This was because at high conservation the high similarity of the ω values creates "noise" for the scoring function and this results in poor alignments. Under such conditions the power of aligning sequences based on selective pressure at amino acid sites becomes weak. Therefore, at high conservation the BLOSUM score is more acceptable than a FIRE score. Conversely, when the amino acid conservation decreases with ω MLEs >0.5 approaching 1 indicating rapid evolution or positive selection, the FIRE score is more acceptable from the biological point of view. Consequently, it was conceptualised to use a BLOSUM score when there is high conservation and use FIRE when conservation was low. To allow the algorithm to be dynamic between the two approaches: BLOSUM to FIRE depending on the selective pressure on the amino acids. Therefore, Equation (3.10) was adjusted to make the

equation dynamic giving:

$$BLOSUM\text{-}FIRE = BLOSUM \times sel_{ab} + K \times sim_{ab}(1 - sel_{ab}) \tag{3.11}$$

The *BLOSUM-FIRE* score of an alignment is the sum of the score in Equation (3.11) normalised by the minimum sequence length. Therefore, each residue position is assigned a *FIRE* score using the Needleman-Wunsch DP algorithm (Section 2.1.1) to obtain the global alignment.

3.6 A variable gap penalty scheme

The affine gap penalty scheme proposed by Gotoh (1982) and Altschul and Erickson (1986) discussed in Section 2.5, was modified to make the insertion of gaps more variable depending on the selective pressure at codon sites. A variable gap penalty scheme based on an affine model was proposed, similar to the form described by Thompson et al. (1994). Furthermore, Equation (3.4) for scaling the BLOSUM score was also adopted for approximating the selective pressure at codon sites to determine the gap penalties. The proposed variable gap penalty uses the same principle for the adjusting BLOSUM scores discussed in Section 3.5.1. Consequently, those sites that are highly conserved have relatively high gap penalties given by the equation:

$$P = ge^{-a} + ele^{-x} \tag{3.12}$$

where P is the total gap open penalty, g is the gap open penalty, e is the gap extension penalty e^{-a} is the selective pressure at the codon site and a is the ω value where the gap is opened, x is the ω value for the site where the gap is extended. Gaps are extended based on the level of conservation at each site and the gap open penalty varies as a function of the ω at each site.

3.6.1 Determining optimal gap penalties

Theoretical guidelines for the determination of gap penalties (Section 2.5) are scarce, in this study an empirical approach similar to the investigation by Reese and Pearson

(2002) was adopted. The datasets consisted of 50 amino acid sequences with their ω MLEs of varying lengths randomly selected from the PFAM-A seed alignments database version 27.0 of protein domains (Bateman, 2004). The FIRE score was used as a proxy for the quality of the alignments, the score is the mean of the sequence identity and the normalised evolutionary rate score (old FIRE score) for that alignment. Gap open penalties were iteratively reduced from 0 to -11, the -11 was chosen because highest residue match score in the BLOSUM matrix is 11. Therefore, PFAM proteins were aligned against each other and the scores for each iteration of the gap open penalty from 0 to -11 were recorded. These experiments were automated using a custom script. Simulations were also carried out with varied gap extension penalties. Iterations were carried out using gap open penalties in the range [0,-11] and gap extension penalties in the range [0,-6]. The rationale is that no significant changes can be detected in alignments when the gap extension penalties are increased above half the maximum gap open penalty (Reese and Pearson, 2002).

3.7 Evaluating the performance of the BLOSUM-FIRE algorithm

Ten datasets were used to evaluate the performance of the algorithm. These datasets consisted of ω profiles of PFAM families with low similarity, the datasets are provided in Table 3.2. The objective was to evaluate the performance of the new scoring function in the BLOSUM-FIRE algorithm by comparing its performance with conventional algorithms including of FIRE algorithm. Corresponding nucleotide sequences were retrieved from the GenBank database and phylogenetic trees for the alignments were retrieved from the PFAM-A database. The CODEML program generated the ω MLEs at codon sites using options described in Section 3.2. In addition to these datasets, domains of the highly conserved κ and λ antibody variable regions were aligned for comparison with other algorithms.

Table 3.2: Domain family datasets used to evaluate the BLOSUM-FIRE algorithm.

Data set	PFAM id	Domain family description
set1	PF05308.6	Mitochondrial fission regulator
	PF14327.1	Hinge domain of cleavage stimulation factor
set2	PF03424.9	Carbohydrate binding domain
	PF07670.9	Nucleoside recognition
set3	PF06938.6	Protein of unknown function
	PF12706.2	Lactamase_B_2
set4	PF09918.4	Uncharacterized protein containing a ferredoxin domain
	PF10832.3	Protein of unknown function
set5	PF13424.1	Tetratricopeptide repeat
	PF01609.16	Transposase domain
set6	PF13922.1	Domain of transcriptional enhancer
	PF02797.10	Chalcone and stilbene synthases
set7	PF08397.6	IRSp53/MIM homology domain
	PF06648.6	Protein of unknown function
set8	PF03064.11	HSV U79 / HCMV P34
	PF07849.6	Protein of unknown function
set9	PF01717.13	Cobalamin-independent synthase catalytic domain
	PF12877.2	Domain of unknown function
set10	PF13704.1	Glycosyl transferase family
	PF07125.6	Protein of unknown function

The BLOSUM-FIRE algorithm software including user information was made available for download at www.bionf.wits.ac.za/software/fire. In addition to the algorithm, sample data files, resources and user information have been made available.

3.8 The development of EvoDB

To infer the function of an unknown protein the first step is to search a sequence database (Section 2.13). The BLOSUM-FIRE algorithm required a database of ω MLEs for the inference of domain functions, therefore the ***Evo***lutionary rates ***D***ata***B***ase (EvoDB) was developed. The PFAM-A database was chosen because it provided a suitable framework for inferring domain function as it contains a comprehensive collection of protein functional domains clustered in families, the PFAM database is described in Section 2.13.1. Challenges arose because of the size of the PFAM database, this led to an automated approach being taken.

The Python programming language was also used to compile the EvoDB because it has very good regular expression support allowing for easy text searching. The clean syntax and dynamic programming infrastructure of Python resulted in rapid program development times. A custom Python script called *dblookup.py* (Appendix C) was created for the specific task of automating the process of retrieving the corresponding coding sequence (CDS) DNA sequences for the PFAM families and determining the ω MLEs profiles using the CODEML program. The script was created to achieve the following:

- Get PFAM sequences and read cross reference mapping information.
- Retrieve amino acid sequences from the local SWISS-PROT database if not found check the TrEMBL database.
- Retrieve GenBank nucleotide sequence accession identifiers and CDS feature information using cross referencing mapping information from the SWISS-PROT protein files.
- Fetch the corresponding nucleotide sequences from the GenBank local database.
- Verify that the nucleic acids sequences are the corresponding CDS for the PFAM protein sequences.
- Run the CODEML program using nucleic acid sequences and corresponding phylogenetic tree files as input and determine the ω MLEs at the codon sites.

3.8.1 Parallelisation of the compilation of EvoBD

The computational resources of the ZA-WITS-CORE (University of the Witwatersrand) cluster of 13 nodes running Scientific Linux 6.3 were utilised to achieve parallelism. The decision to use the cluster was due to the fact that the retrieval of sequences and the determination of the ω MLE profiles using CODEML was computationally expensive. Given the size of the PFAM database of 13672 families, this would not have been feasible on a desktop machine. Figure 3.4 shows the *dblookup.py* pipeline and how

the different programs were utilised to generate the corresponding nucleotide sequences and subsequently the calculation of the MLEs of ω using the CODEML program.

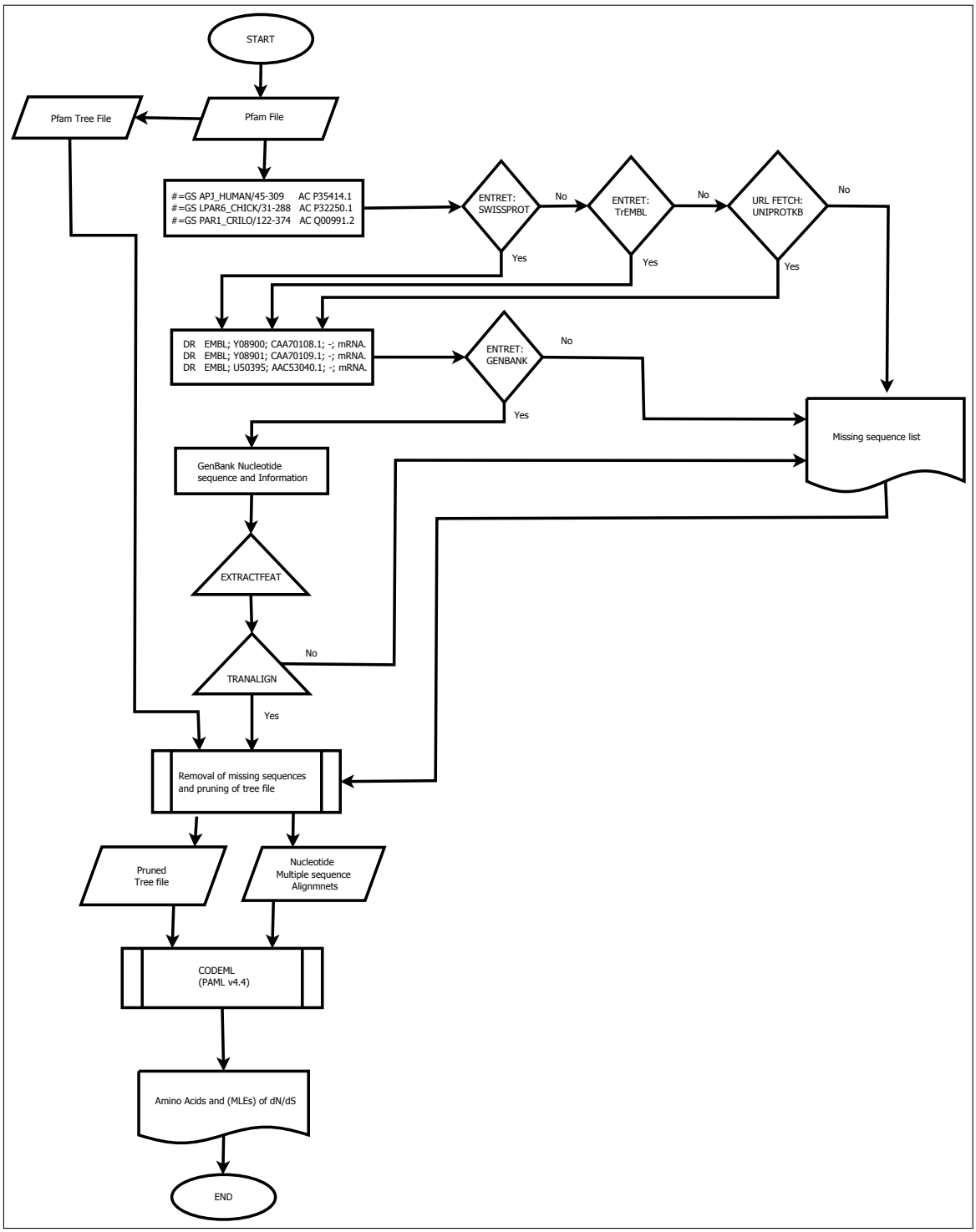


Figure 3.4: The EvoDB compilation pipeline, the process begins with a PFAM stockholm family file. Using utilities from EMBOSS, sequences and cross reference information are used to obtain nucleotide and trees files for the PFAM family.

Given the embarrassingly parallel nature of the task of compiling the EvoDB; a parallel approach was chosen. The cluster PBS (Portable Batch System) scheduling system submitted the computation tasks called jobs on the ZA-WITS-CORE cluster. Computational resources were managed by the MAUI/TORQUE (***T***erascale ***O***pen-source ***R***esource and ***QUE***ue Manager) resource manager. A secondary custom Python scheduler was written to manage the submission of jobs, this added more control over the number jobs submitted as each PFAM family was submitted separately as a single job. A total of 13672 jobs were submitted and the maximum number of parallel jobs at a time was capped at 80 due to resource limits. The secondary scheduler was also used to monitor the jobs queue and to ensure the maximum number of jobs submitted at a time was not exceeded. Figure 3.5 provides a shell script used to submit jobs on the PBS system.

```
#!/bin/bash
#PBS -N EvoDBX
#PBS -l nodes=1:ppn=3,walltime=300:00:00,mem=3GB
#PBS -q WitsLong
python  dblookup.py  $family
```

Figure 3.5: A shell script used to submit the tasks on the Wits cluster using the *qsub* command on the cluster. The *$family* variable is passed in as an argument from the command line.

3.8.2 Retrieving sequence information from the PFAM database

The PFAM-A seed alignment database version 27.0, GENBANK version 193.0, SWISS-PROT and TrEMBL databases accessed in January 2013 were made locally available on the cluster. A custom Python script spliced the PFAM database file (PFAM-A.seed) and created local directories for each PFAM family. The *dblookup.py* program submitted each family to the PBS system on the ZA-WITS-CORE cluster, each family was submitted as a single job on the PBS cluster system as described in Section 3.8.1. Python regular expressions were used to extract the SWISS-PROT cross reference information from PFAM stockholm files as shown on Figure 3.6, this information was then utilised for the retrieval of sequences from the local SWISS-PROT database.

Figure 3.7 shows an example of a line containing the generic per-sequence annotation where regular expressions were used to extract the SWISS-PROT accession numbers from the PFAM Stockholm file.

```
//
# STOCKHOLM 1.0
#=GS P53_DANRE/63-257      AC P79734.1
#=GS P53_ONCMY/83-278      AC P25035.1
#=GS P53_HUMAN/95-289      AC P04637.4
#=GS P53_HUMAN/95-289      DR PDB; 3Q06 B; 96-289;
#=GS P53_HUMAN/95-289      DR PDB; 2VUK A; 95-289;
#=GS P53_HUMAN/95-289      DR PDB; 1KZY B; 95-289;
#=GS P53_HUMAN/95-289      DR PDB; 2QXB A; 95-289;
#=GS P53_HUMAN/95-289      DR PDB; 2PCX A; 95-289;
#=GS P53_HUMAN/95-289      DR PDB; 2X0U B; 95-289;
P53_HUMAN/95-289                SSVPSQKTYQGSYGFRLGFLH.SGTAKSVTCTYSPALNKMFCQL.....
#=GR P53_HUMAN/95-289      SS   SSS---S-------EE-----.----TTTSEEEETTTTEEEEET.....
P53_XENLA/69-264                CAVPSTDDYAGKYGLQLDFQQ.NGTAKSVTCTYSPELNKLFCQL.....
P53_ORYLA/80-270                TTVPVTTDYPGSYELELRFQK.SGTAKSVTSTYSETLNKLYCQL.....
Q27937_LOLFO/120-314            PSVPSNIKYPGEYVFEMSFAQPSKETKSTTWTYSEKLDKLYVRM.....
P73_HUMAN/113-309               PVIPSNTDYPGPHHFEVTFQQ.SSTAKSATWTYSPLLKKLYCQI.....
#=GR P73_HUMAN/113-309     SS   ---------------EEE---.---STT-SEEEETTTTEEEE-T.....
#=GC SS_cons                    SSS---S-------EEEE---.---STTTSEEEETTTTEEEEET.....
#=GC seq_cons                   soVPossDYPGsasFcLpF.Q.SuTAKSVTsTYSPpLNKLaCQL.....
//
```

Figure 3.6: A truncated PFAM file for the P53 family showing the database cross-reference mapping information for the SWISS-PROT proteins found in the P53 family.

3.8.3 Retrieval of protein files from SWISS-PROT and TrEMBL

SWISS-PROT accession numbers were used to retrieve sequences from the SWISS-PROT database. The ENTRET program in EMBOSS (Rice et al., 2000) was used to retrieve the sequences from the local database. The ENTRET program retrieves sequence entries from local databases using accession numbers provided as parameters. Those sequences that could not be found in the SWISS-PROT database were searched for in the computer annotated TrEMBL, otherwise they were added to the missing list.

```
#=GS P53_DANRE/63-257      AC P79734.1
```

Figure 3.7: A line extracted from a PFAM stockholm file showing the line that contains reference information that is required to fetch the SWISS-PROT protein file, in this case it would be P79734.1. This line was extracted from the file in Figure 3.6.

3.8.4 Extracting corresponding nucleotide sequences from GenBank files

To compile EvoDB, database cross reference information from the SWISS-PROT protein files as shown on Figure 3.8 was utilised to access nucleotide sequence files from the GenBank database. Next, nucleotide sequences were retrieved from the local GenBank database using the ENTRET program from the EMBOSS suite of software. The EXTRACTFEAT program from EMBOSS extracts the CDS regions that code for a protein of interest, the program uses the annotation information to identify the CDS regions in a GenBank file. Therefore, corresponding CDS for PFAM proteins were retrieved from the GenBank local database using the EXTRACTFEAT program. Those sequences that could not be found on the GenBank database were added to the missing list.

```
DR   EMBL; S78694; AAB21243.1; -; mRNA.
DR   EMBL; M94054; AAA59525.1; -; mRNA.
DR   EMBL; S45875; AAB23549.1; -; mRNA.
DR   EMBL; AF039291; AAD02130.1; -; mRNA.
DR   EMBL; AK312269; BAG35200.1; -; mRNA.
DR   EMBL; BC074820; AAH74820.1; -; mRNA.
DR   EMBL; BC074872; AAH74872.1; -; mRNA.
DR   EMBL; BC089436; AAH89436.1; -; mRNA.
DR   EMBL; L16895; AAA16870.1; -; Genomic_DNA.
DR   EMBL; M84150; AAA59541.1; -; Genomic_DNA.
```

Figure 3.8: A truncated SWISS-PROT file. The annotation information contains cross reference mapping for GenBank and feature information where the coding sequences can be retrieved.

The TRANALIGN utility from EMBOSS uses an algorithm to validate the corresponding nucleotide coding sequences of an aligned protein. It does this by translating the provided nucleotide sequence and determines using three forward nucleotides, frame by frame if the coding sequences correspond to the aligned protein sequence. The TRANALIGN utility was used on extracted CDS and the PFAM proteins files to ensure that the nucleotide sequences extracted were indeed the CDS responsible for the

PFAM protein. The TRANALIGN utility validated and quality controlled sequences to verify and prevent mis-annotations in the SWISS-PROT files and GenBank files resulting in erroneous CDS in the alignments. Those PFAM sequences whose alignments could not be validated using TRANALIGN were added to the missing list.

3.8.5 Evaluating the efficacy of pruning phylogenetic trees

Phylogenetic trees for each PFAM family, available on the PFAM database, were utilized for generating the ω MLEs for the PFAM sequences. A decision was made to prune the existing phylogenetic trees for those sequences that could not be retrieved. To prune files required less computational resources than generating phylogenetic tree using the MSA of nucleotide sequences. The large number of sequences in each PFAM database would require prohibitive CPU times that scaled with the number of sequences. Trees generated using the FastTree algorithm available from PFAM were used. Nucleotide sequences that could not be found in the alignment were pruned using the Prune function in the Bio.Nexus class found in the Biopython module (Cock et al., 2009). This function removed those nodes in the tree files whose nucleotide sequences could not be retrieved from GenBank or found in the missing list compiled by the *dblookup* utility as shown in Figure 3.4. The pruning algorithm removes nodes and its branch is added to the remaining node. The efficacy of utilising the pruning algorithm instead of generating phylogenetic trees was evaluated.

To evaluate the efficacy of pruning against calculating trees from sequences, a tree of 35 nodes was randomly selected and 30 nodes were pruned from the tree using the Prune function in the Bio.Nexus class as used in the *dblookup* utility. The protein sequences used to generate this tree were obtained and those corresponding nodes that were removed in the tree file were removed. The amino acid sequences were then used to generate a phylogenetic tree using FastTree version 2.1.7 (Price et al., 2010), the algorithm used for calculating phylogenetic trees in the PFAM database. Those

generated phylogenetic trees were then compared and their qualities evaluated.

3.8.6 Compiling EvoDB

The posterior probabilities of BEB of the MLEs of ω of the codon sites were extracted from the *rst* files produced by the CODEML program. A custom Python script compiled all the PFAM ω MLEs into a flat file database for all the representative families. The compilation of EvoDB included the corresponding pruned phylogenetic trees in newick format for the ω profiles, gapped nucleotide alignments in nexus format and the corresponding PFAM alignments in stockholm format. Additionally, annotated data in the form of accession identifiers for all the sequences including those sequences that could not be retrieved due to errors and miss-annotation issues is provided on the database. The utility of EvoDB as a general purpose resource was also realised, the database has been made available for download including release notes at www.bionf.wits.ac.za/software/fire/evodb. In addition to the database, a script for retrieving data from the database is also available, external links to resources related to EvoDB have also been provided.

3.9 Searching EvoDB using the HBx protein ω profile

HBx curated nucleotide sequences were obtained from Thompson (2012), who carried out an evolutionary analysis of the protein. HBx was chosen because the inference of its functions has been elusive (Section 2.15). An alignment of 20 nucleotide sequences of the HBx were aligned using the CLUSTAL Omega program and the phylogenetic trees were generated using the FastTree program. Next, ω MLEs were determined using the CODEML program using options and parameters described in Section 3.2. The HBx domain protein was aligned against all of the PFAM families in EvoDB using a custom script. All the alignment scores and results of the alignment were sorted and saved as Python pickled dictionary objects, this ensured fast search execution times.

3.9.1 The statistical significance of the results

To evaluate the significance of the results a statistical framework was required, Section 2.7 describes the theoretical framework for the statistical significance of alignments. A p-value framework was implemented to assess the statistical significance of the results. HBx alignment scores on EvoDB were recorded and sorted according to statistical significance. The HBx ω MLEs were shuffled using a function available in the algorithm to query the EvoDB database. The scores of the alignments were tested for statistical significance using a p-value statistical framework. The fraction of those scoring higher than the actual alignment was obtained and this provided the p-value of that alignment.

3.9.2 Assessing the inferred function of the HBX

The functions of those proteins that were statistically significant were assessed by first analysing the alignments produced. The BLAST algorithm finds statistically significant regions of local similarity between two sequences. The results were then assessed using the BLAST algorithm (Altschul et al., 1990) at `www.ncbi.nlm.nih.gov/blast` to assess their statistical significance. A literature search was also utilised to investigate if these inferred functions have been documented and to confirm the biological relevancy of the results. Furthermore, the alignment produced by BLOSUM-FIRE was queried for motifs using the ***M***ultiple ***EM*** for ***M***otif ***E***licitation (MEME) (Bailey and Elkan, 1994) on-line web server at `http://meme.nbcr.net/meme/`. While similar motifs are not an indicator of shared function, they can highlight structural similarities between two proteins especially when the structures are not available.

Chapter 4

Results

The salient results of experiments described in the methods chapter are presented in this chapter. The first results presented here were obtained from evaluating the FIRE algorithm. Those datasets where the FIRE algorithm produced false positive results were identified and their alignments analysed. The results from this analysis were then used to investigate how to improve the algorithm alignment sensitivity and scoring function specificity. The chapter provides the results of the investigation used in the conceptualisation and implementation of a new algorithm, BLOSUM-FIRE, where two approaches: one based on the evolutionary rate and the other a substitution matrix, were combined. Included here also, are the results of the evaluation of the efficacy of coupling the FIRE approach with a BLOSUM approach. The results of the implementation of the Evolutionary rate Database (EvoDB) are presented. Finally, the results of the alignment of the HBx protein against the profiles of EvoDB and results of assessing the HBx results conclude the chapter.

4.1 Evaluation of the FIRE algorithm using real data

The FIRE algorithm was evaluated to identify the source of the false positives and to identify the limitations of the evolutionary rate approach as described in Section

3.3. The results presented here were obtained using real data sets, these were also used in the concept paper of Durand et al. (2010). In these experiments the datasets were aligned against each other and their alignments and scores were analysed. Those data sets that resulted in false positives were identified, their alignments and data were analysed.

Table 4.1: Aligned sequences and their scores.

Aligned sequences	***FIRE score**
Protozoa MYB1 and Metazoan MYB1	0.94
Kappa and Lambda	0.71
Metazoa MYB1 and Lambda	0.57
Protozoa MYB1 and Lambda	0.53
Kappa and metazoan MYB1	0.51
Protozoa MYB1 and P53DBD	0.50
Protozoa MYB1 and Kappa	0.48
P53DBD and Metazoan MYB1	0.47
P53DBD and Lambda	0.28
GK and Metazoan MYB1	0.09
GK and Lambda	0.03
Metazoan MYB1 and Lambda	0.03

*The FIRE score is calculated using the similarity of the ω at codon sites normalised in the range [0,1] using the maximum sequence length. P53DBD is a tumor suppressor DNA-binding-domain, GK is a metabolic enzyme glycerol kinase and Lambda and Kappa are light chain antibody variable regions. Protazoa and Metazoa MYB1 are ortholog transcription factors.

The results (Table 4.1) show the salient results from the evaluation of the FIRE algorithm using some of the data sets used by Durand et al. (2010). These results show the FIRE scores for functionally related and functionally unrelated sequences. Using the guidelines provided by Durand et al. (2010) a 60% (0.6 FIRE score) minimum threshold was used to infer those domains with similar functions according to the FIRE hypothesis. The results show a high score for the orthologous protozoa MYB1 and metazoa MYB1. The Kappa and Lambda also produced relatively high FIRE scores indicating that they share similar functions. Therefore, these results show that the algorithm is able to detect those domains with shared functions and more importantly they demonstrate the utility of the FIRE approach in inferring domain functions (Table

4.1).

On the other hand, some of these results are false positive for example the Metazoa MYB1 and Lambda pair do not share similar functions, but have relatively high FIRE scores. Table 4.2 displays the statistical analysis of the ω MLE in the data used to generate the alignments presented in Table 4.1.

Table 4.2: Analysis of the distribution of the ω MLE values for the data sets used to evaluate the algorithm.

Data set	Min	Max	Median	Variance	Mean
Kappa	0.05	0.054	0.05	2.2×10^{-07}	0.05
P53DBD	0.05	1.8	0.05	0.14	0.22
GK	0.2	3.5	0.51	0.28	0.69
Metazoa MYB1	0.65	7.0	1.6	3.2	2.5
Protozoa MYB1	0.05	2.4	0.05	0.061	0.098
Lambda	0.0	2.1	0.036	0.16	0.22

To further investigate the occurrence of false positives the performance of the FIRE algorithm was compared to two conventional algorithms: MAFFT and CLUSTAL Omega scores using a 100 randomly selected families from the PFAM-A database as described in Section 3.3. The results in Figure 4.1 demonstrate the divergence between the FIRE algorithm identity score distribution compared to conventional algorithms. The FIRE algorithm scores are very high however the alignments are poor with lower sequence identity scores.

4.2 Evaluation with simulated data sets

The results of evaluating the algorithm with real data sets provided the information used to generate simulated datasets. Data sets were simulated as described in Section 3.3.1, one of the datasets used in this study is provided in Figure 3.2.

The performance of the FIRE algorithm was compared with CLUSTAL Omega and MAFFT for simulated highly conserved datasets to demonstrate false positive results and limitations of the FIRE algorithm. Figure 4.3 shows the CLUSTAL Omega,

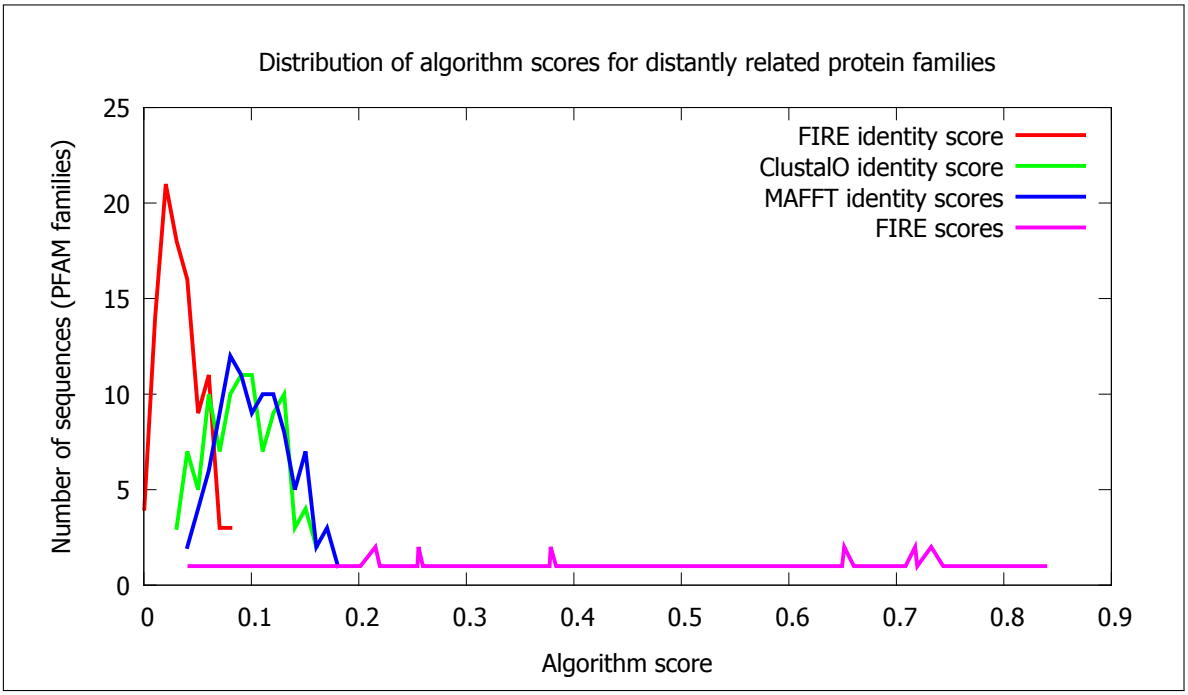


Figure 4.1: Comparison of the FIRE algorithm with CLUSTAL Omega and MAFFT algorithms. Identity scores were calculated by normalising the number of matched residues with the maximum sequence length. The FIRE score is the algorithm score for the aligned sequences based on the evolutionary rate approach.

MAFFT and T-COFFEE alignment using default parameters for the same pair of sequences aligned by the FIRE algorithm in Figure 4.2. The quality of alignments obtained and number of residues matched from each algorithm were compared. The results revealed that for the simulated sequences the CLUSTAL Omega algorithm was able to produce 2 residue matches and inserted gaps where the FIRE algorithms did not.

```
9.99066666667
0.999066666667
>testseqx1.txt
GTAKSVMCTY
>testseqx2.txt
KGGAWKNTED
```

Figure 4.2: Fasta alignment produced by the FIRE algorithm for simulated highly conserved sequences, the first line in the output indicates the normilsed score of ω matches and the second line is the FIRE score.

CLUSTAL Omega alignment

```
testseqx1    --GTAKSVMCTY
testseqx2    KGGAWKNTED--
```

MAFFT alignment

```
testseqx1   --GTAKSVMCTY
testseqx2   KGGAWKNT--ED
               *: *..
```

T-COFFEE alignment

```
testseqx1       --GTAKSVMCTY
testseqx2       KGGAWKNTED--
                  *: *..
```

Figure 4.3: T-COFFEE, CLUSTAL Omega and MAFFT alignments for simulated highly conserved sequences, compare with Figure 4.2 for the FIRE algorithm with a score of 0.9991 with no residues matches or gaps inserted.

4.3 Comparison of the quality of alignments produced by FIRE and BLOSUM matrix based approaches

Three different data sets, described in Section 3.4.1, were chosen to evaluate and compare the quality of alignments produced by the two approaches: FIRE and the conventional BLOSUM matrix based approach. The quality of alignments produced by the two were analysed and the results are presented in Table 4.3. In addition, these results demonstrate that the FIRE algorithm produced false positive in highly conserved data sets: P53 and a paralog of metazoa MYB. The concept of a BLOSUM and FIRE coupled approach was formulated by analysing these results.

Table 4.3: Comparison of the quality of alignments produced by the FIRE algorithm and BLOSUM62 matrix based algorithm.

Data Set	FIRE score	FIRE *ID	BLOSUM *ID
Light chain antibody κ and antibody λ genes (Highly conserved sharing similar functions).	0.71	0.23	0.45
P53 DBD and paralog metazoa MYB (Highly conserved unrelated sequences).	0.47	0.02	0.17
P53 and antibody κ (Unrelated and varied conservation).	0.28	0.04	0.21

*ID is the identity score for the alignments given by number of positional amino acids residues matched normalised by the maximum length of the two sequences to a range $[0, 1]$ for comparison with FIRE scores.

4.3.1 Alignment of highly conserved sequences

The λ light chain and κ Light chain antibody variable region genes are highly conserved datasets with a median ω in the range $[0.0036, 0.05]$. These analogous genes have been found to share similar functions, however, the have a low sequence similarity with sequence identity in the 20% to 25% range, the statistical analysis of the distribution of ω values is provided (Table 4.2).

The alignment produced byλ light chain and κ Light chain variable regions generated by the FIRE algorithm is shown in Figure 4.4. The sequences in Figure 4.4 were aligned using a BLOSUM62 based algorithm and the alignment is provided in Figure 4.5. Comparison of alignments displayed in Figure 4.4 and Figure 4.5 and results in Table 4.3 revealed that a BLOSUM based approach was more effective at aligning residues compared to a FIRE based approach. The FIRE score (Table 4.3) for the alignments can be used to infer that the two sequences are responsible for the same

```
Lambda    VSGSPGKTITISCSGSDIATTNIQWYQQRPGSAPTAVIYEDNRRPS...G
Kappa     FL.SASVGDRVTITCRAS.QGISSLAWYQQKPGKA.PKLLIYAASTLQSG

Lambda    VPDRISGTISSNSASLTISDLKTEDEAEYYCQ..A
Kappa     VPSRFSGSGSGTEFTLTISSLQPEDFATYYCQQLN
```

Figure 4.4: FIRE alignment for highly conserved sequences: λ light chain and κ Light chain antibody variable regions.

```
Lambda.txt    .VSGSPGKTITISCSGSD.IATTNIQWYQQRPGSAPTAVIYEDNRRPSGV
Kappa.txt     FLSASVGDRVTITCRASQGISS..LAWYQQKPGKAPKLLIYAASTLQSGV

Lambda.txt    PDRISGTISSNSASLTISDLKTEDEAEYYCQA..
Kappa.txt     PSRFSGSGSGTEFTLTISSLQPEDFATYYCQQLN
```

Figure 4.5: BLOSUM62 based algorithm alignment for λ light chain and κ Light chain antibody variable regions.

function according to the FIRE hypothesis. On the other hand, the BLOSUM based algorithm obtained more residues matches resulting in a higher ID score than the FIRE algorithm. However, FIRE also produced more gaps than the BLOSUM approach and analysis of these gaps revealed that most of them correspond to sites of positive selection with $\omega > 1$. Furthermore, there was a correlation between the number of gaps and number of sites with $\omega > 1$. This was attributed to the scoring function which assigns a zero score when the difference between two ω values is greater than the parameterized cap of ω_{max} of 1.5 described in Section 3.1 as shown by Equations (3.7) and (3.8).

4.3.2 Alignments of highly conserved unrelated sequences

Evaluation of the scores and alignments produced by the FIRE algorithm revealed that there was a high rate of false positives in highly conserved datasets. The FIRE approach and BLOSUM62 matrix based approach were evaluated using highly conserved unrelated data sets with relatively high sequence length divergence: P53 DBD and Metazoan MYB1. The results (Table 4.3) and alignments for FIRE (Figure 4.6) and

```
Metazoa    KGGAWKNTEDAV.....SKYGKNQWARI.....................
P53        GTAKSVMCTYSPPLNKLFCQLAKTCPVQLWVSATPPAGSRVRAMAIYKKS

Metazoa    .SSLLVRKTPKQCKA..................................
P53        QHMTEVVRRCPHHERCSDGDLAPPQHLIRVEGNLYPEYLEDRQTFRHSVV

Metazoa    .....RWYEWIDPSIKKTEWSREEDEKLLHLAKLLPTQWRT.........
P53        VPYEPPEAGSEYTTIHYKYMCNSSCMGGMNRRPILTIITLEDSSGNLLGR

Metazoa    ..IAPIVGRTATQCLERYQ
P53        DSFEVRVCACPGRDRRTEE
```

Figure 4.6: FIRE alignment of highly conserved unrelated sequences: P53 DBD and Metazoan MYB1. The shorter sequence has been stretched over the length of the longer sequence resulting in gaps, a result of using the global Needleman-Wunsch algorithm.

```
Metazoa    KGGAWKN...T.EDAVSK.Y...GKN...Q.W......A..RISSL....
P53        .GTA.KSVMCTYSPPLNKLFCQLAKTCPVQLWVSATPPAGSRVRAMAIYK

Metazoa    .......LVRKTP..KQCKARWYEWID...PS..IKKTEWSR..E..EDE
P53        KSQHMTEVVRRCPHHERCS....DG.DLAPPQHLIR.VEGNLYPEYLEDR

Metazoa    KLL.HLAKLLP.......TQWRTI.................APIV......
P53        QTFRH.SVVVPYEPPEAGSEYTTIHYKYMCNSSCMGGMNRRPILTIITLE

Metazoa    .......GRTATQ...CL....ERY..Q
P53           DSSGNLLGRDSFEVRVCACPGRDRRTEE
```

Figure 4.7: BLOSUM62 matrix based algorithm alignment for highly conserved functionally unrelated sequences: P53 DBD and Metazoan MYB1. The alignment is similar to the alignment produced by the FIRE algorithm. Compared to the FIRE algorithm in Figure 4.6 more gaps were inserted and more amino acid residues aligned.

the BLOSUM62 based approach (Figure 4.7) demonstrate how the Needleman-Wunsch global alignment algorithm generates stretched alignments in cases of sequence length divergence as discussed in Section 2.1.1.

4.3.3 Alignments of unrelated sequences with variable conservation

The unrelated sequences of P53 DBD and a κ antibody variable region were aligned to compare the efficacies of the FIRE and BLOSUM matrix based approaches. The FIRE and BLOSUM alignments are shown in Figures 4.8 and 4.9 respectively. The analysis of their alignment is presented in Table 3.4.1, with almost similar percentage identity scores of 0.28 for FIRE and a BLOSUM62 based algorithm identity score of 0.21 demonstrating the low sequence relatedness between the sequences hence functionally unrelated. The results also show that the FIRE score and BLOSUM62 algorithm score converge when sequences have variable conservation. The sequences had a relatively high sequence length divergence and this resulted in gapped alignments, however, this is expected of global alignments as they find similarities that span the whole length of a sequence. In addition, the arrangement of gaps was typical of FIRE, long gaps interspersed by long blocks of amino acids.

```
Lambda    VS.........GSPGKTI....TISCSGSDIAT..TNIQ.........
P53_DBD   GTAKSVMCTYSPPLNKLFCQLAKTCPVQLWVSATPPAGSRVRAMAIYKKS

Lambda    ..............WYQ.QRPGSA..........PTAVIYEDNRRPSG.
P53_DBD   QHMTEVVRRCPHHERCSDGDLAPPQHLIRVEGNLYPEYLEDRQTFRHSVV

Lambda    ..VPDRISGTISSNSASLTI.....................SDLKTEDEA
P53_DBD   VPYEPPEAGSEYTTIHYKYMCNSSCMGGMNRRPILTIITLEDSSGNLLGR

Lambda    EY............YCQA
P53_DBD   DSFEVRVCACPGRDRRTEE
```

Figure 4.8: Alignment produced by the FIRE algorithm for P53 DBD and antibody κ variable region. The alignment shows the effect of sequence length divergence in a global alignment algorithm. Once again, the FIRE algorithm produced fewer gaps with blocks of residues along the length of the alignment.

```
Lambda    VSGSPGKTI..TIS.......CSGSDIATT.NIQ.WYQQRP..GSAPTAV
P53_DBD   ..GT.AKSVMCTYSPPLNKLFCQ...LAKTCPVQLWVSATPPAGSRVRAM

Lambda    .IYEDN.......RR.P.....S.G..........V.....P....DR..
P53_DBD   AIYKKSQHMTEVVRRCPHHERCSDGDLAPPQHLIRVEGNLYPEYLEDRQT

Lambda    ......I......SG....TI.......SS.....NSAS.LTI...SD..
P53_DBD   FRHSVVVPYEPPEAGSEYTTIHYKYMCNSSCMGGMNRRPILTIITLEDSS

Lambda    ..LKTEDEAEY.YC.........QA
P53_DBD   GNLLGRDSFEVRVCACPGRDRRTEE
```

Figure 4.9: BLOSUM62 matrix based algorithm alignment for P53 DBD and antibody κ variable region. The number of residues matched is relatively similar to the FIRE approach however, the insertion of gaps is different.

4.4 Approximating the selective pressure and scaling the BLOSUM score

The results of the evaluation of the two approaches presented in the previous section was used to conceptualise an approach to couple the FIRE scoring function with a BLOSUM matrix based scoring function. It was also conceptualised to take advantage of the information presented by the ω parameter as an indicator of selective pressure to scale the BLOSUM score. Equation (3.4) described in Section 3.5.1 was proposed to approximate the relationship between the total of the ω values and selective pressure (sel_{ab}) at aligned amino acid sites.

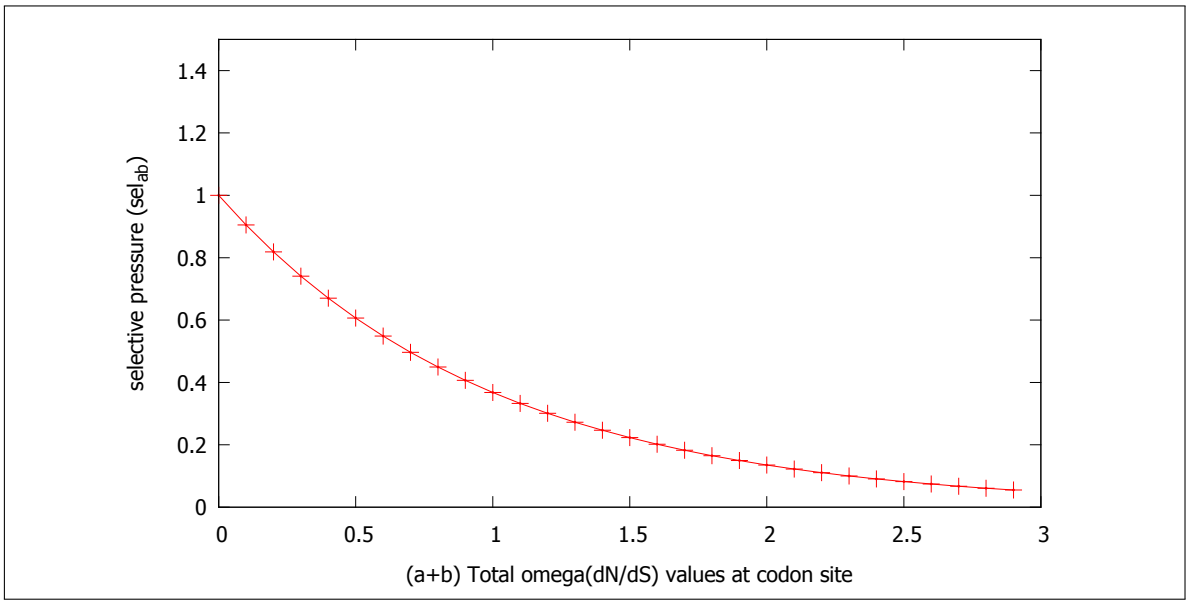


Figure 4.10: Approximating the selective pressure at codon sites using the sum of the evolutionary rates (d_N/d_S) at codon sites: a and b and the approximated selective pressure (sel_{ab}). A total ω value of 0 represents the highest level of selective pressure.

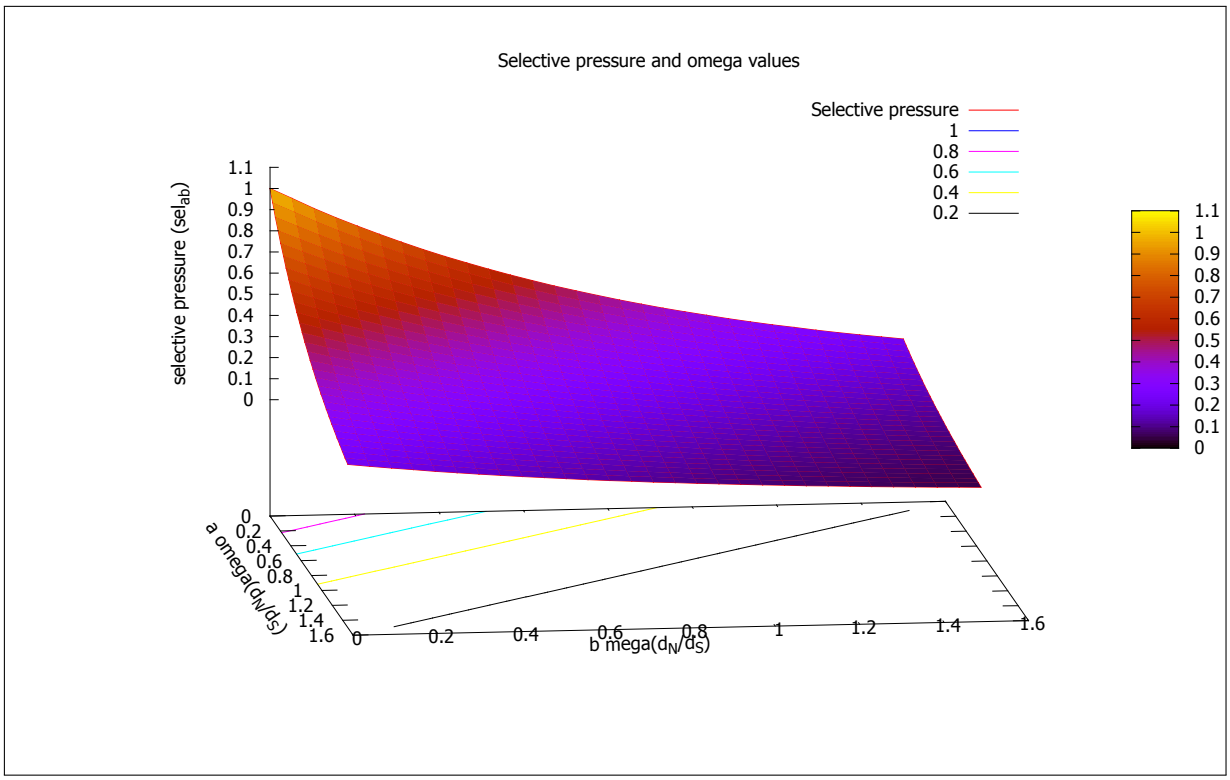


Figure 4.11: Three dimensional heat map for the approximation of selective pressure (sel_{ab}) at sites using ω values at sites: a and b. The selective pressure decreases as the values of ω increase. Equation (3.4) provides the equation for determining sel_{ab} .

This relationship is illustrated in Figure 4.10 showing the general relationship between the sum of evolutionary rates: a and b at codon sites. In Figure 4.11 a three dimensional heat map provides the detailed correlation between ω values: a and b and the approximated selective pressure (sel_{ab}).

Selective pressure was also used to scale the BLOSUM score using Equation (3.6). The impact of the BLOSUM score decreases with increased ω value as the selective pressure (sel_{ab}) on the amino acids decrease. Therefore, the selective pressure on the codon site decreases as the selective pressure moves from negative selection ($\omega < 1$) to positive selection ($\omega > 1$). Equation (3.6) was used to scale a BLOSUM score and the graph in Figure 4.12 was produced to demonstrate the negative correlation between a BLOSUM score and total ω at aligned amino acid site. The figure shows the inverse relationship between sel_{ab} and ω values at sites: a and b at codon site for amino acids with a BLOSUM score of 5, for example, amino acids: A and A.

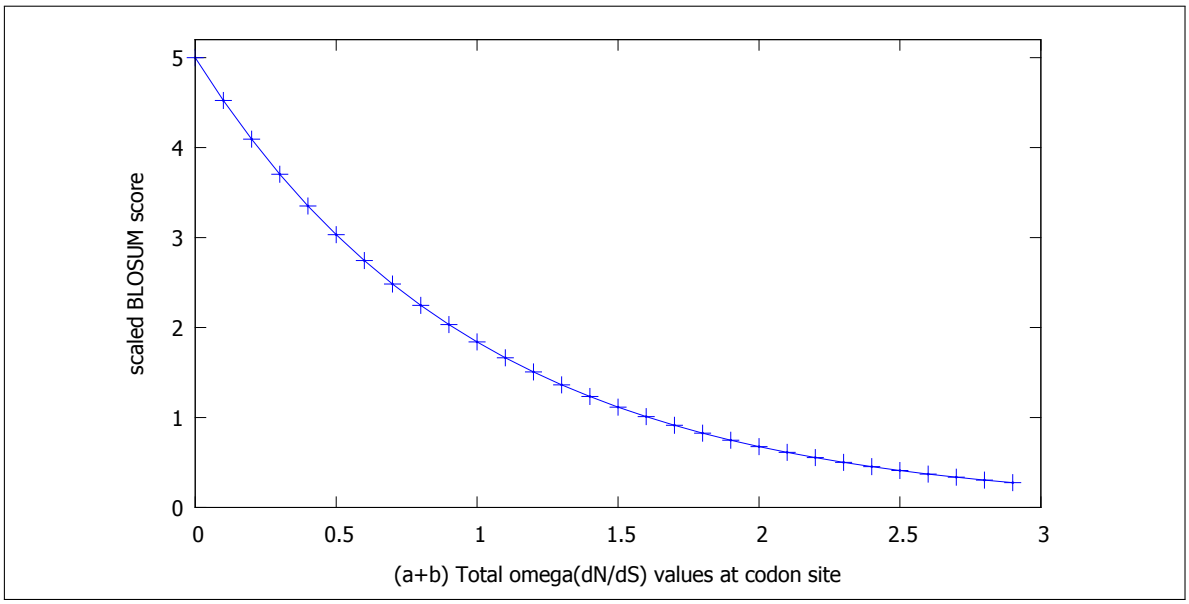


Figure 4.12: How a pair of aligned residues with a BLOSUM matrix score of 5 are scaled. The scaling of a BLOSUM score using the sum of d_N/d_S (ω) values at codon sites as an indicator of selective pressure using the proposed approximation provided in Equation (3.6).

The FIRE algorithm finds similarities between amino acid sequences, a three di-

mensional plot using simulated data is provided to illustrate how the algorithm scores aligned amino acids sites using the values of ω.

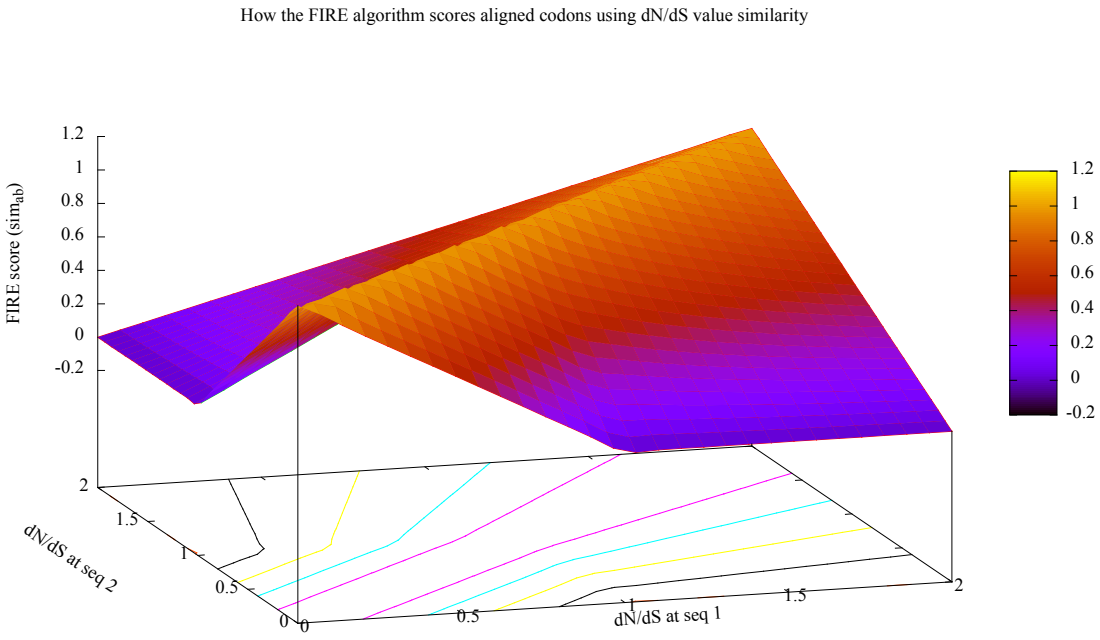


Figure 4.13: How a pair of aligned sequences: seq 1 and seq 2 are are scored using the dN/dS values at codon sites. The more similar the d_N/d_S values the higher the $similarity$ score given by sim_{ab}.

Therefore, the cost function, sim_{ab}, generates alignments by maximising the similarity metric between any two aligned codon sites as shown on Figure 4.13.

4.5 Determining the proportionality constant

The proportionality constant (K) is a scaling factor used to scale FIRE scores to BLOSUM comparable values. To be able to couple the two approaches, a proportionality constant was proposed. The proportionality constant is the mean of the identical amino acid BLOSUM match scores. Identical amino acid match scores were used to produce Figure 4.14.

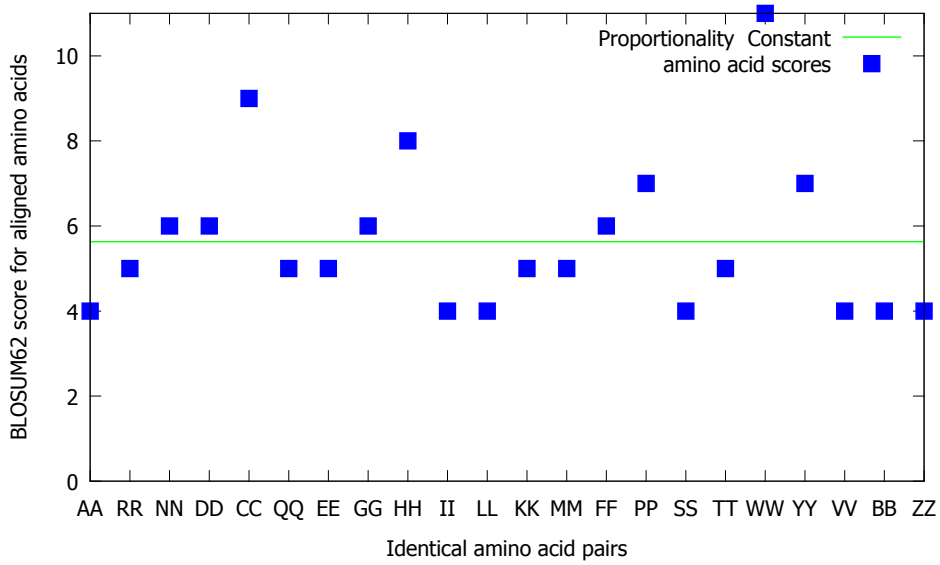


Figure 4.14: The distribution of the scores for identical amino acid matches indicating the location of the proportionality constant ($K = 5.63$) for the BLOSUM62 matrix indicated by the green line.

4.6 Performance of the BLOSUM-FIRE algorithm

The performance of the newly conceptualized BLOSUM-FIRE algorithm was compared with that of conventional widely used algorithms using datasets of low sequence similarity as described in Section 3.7, the results are provided in Table 4.4.

Table 4.4: Comparison of the BLOSUM-FIRE performance with conventional widely used algorithms[†].

Data set	T-COFFEE	MUSCLE	MAFFT	*B-FIRE	FIRE	CLUSTALO
set1	0.10	0.07	0.12	0.09	0.04	0.07
set2	0.11	0.13	0.13	0.17	0.01	0.06
set3	0.10	0.09	0.11	0.16	0.04	0.08
set4	0.12	0.14	0.13	0.20	0.04	0.13
set5	0.06	0.07	0.07	0.10	0.03	0.05
set6	0.13	0.15	0.14	0.16	0.03	0.14
set7	0.11	0.10	0.12	0.11	0.02	0.07
set8	0.06	0.07	0.07	0.10	0.02	0.06
set9	0.12	0.12	0.10	0.12	0.03	0.12
set10	0.10	0.13	0.07	0.15	0.03	0.06

[†]Numbers provided represent the identity score calculated from the number residues matched normalised by the maximum sequence length. *B-FIRE is the new BLOSUM-FIRE algorithm.

CLUSTAL Omega alignment

```
Kappa.txt       -VSGSPGKTITISCSGSDIATTNIQWYQQRPGSAPTAVIYEDNRRPSGVPDRISGTISSN
Lambda.txt      FLSASVGDRVTITCRASQ-GISSLAWYQQKPGKAPKLLIYAASTLQSGVPSRFSGSGSGT
                 :*.* *. :**:* .*: . :.: ****:**.**. :**  .   ****.*:**: *..

Kappa.txt       SASLTISDLKTEDEAEYYCQA--
Lambda.txt      EFTLTISSLQPEDFATYYCQQLN
                . :****.*: ** * ****
```

New FIRE (BLOSUM-FIRE) alignment

```
Kappa.txt       .VSGSPGKTITISCSGSDIATTNIQWYQQRPGSAPTAVIYEDNRRPSGVP
Lambda.txt      FLSASVGDRVTITCRASQGISSLA.WYQQKPGKAPKLLIYAASTLQSGVP

Kappa.txt       DRISGTISSNSASLTISDLKTEDEAEYYCQ..A
Lambda.txt      SRFSGSGSGTEFTLTISSLQPEDFATYYCQQLN
```

BLOSUM62 matrix based algorithm alignment

```
Lambda.txt    .VSGSPGKTITISCSGSD.IATTNIQWYQQRPGSAPTAVIYEDNRRPSGV
Kappa.txt     FLSASVGDRVTITCRASQGISS..LAWYQQKPGKAPKLLIYAASTLQSGV

Lambda.txt    PDRISGTISSNSASLTISDLKTEDEAEYYCQA..
Kappa.txt     PSRFSGSGSGTEFTLTISSLQPEDFATYYCQQLN
```

Old FIRE alignment

```
Kappa.txt       VSGSPGKTITISCSGSDIATTNIQWYQQRPGSAPTAVIYEDNRRPS...G
Lambda.txt      FL.SASVGDRVTITCRAS.QGISSLAWYQQKPGKA.PKLLIYAASTLQSG

Kappa.txt       VPDRISGTISSNSASLTISDLKTEDEAEYYCQ..A
Lambda.txt      VPSRFSGSGSGTEFTLTISSLQPEDFATYYCQQLN
```

MAFFT alignment

```
Kappa.txt       -VSGSPGKTITISCSGSDIATTNIQWYQQRPGSAPTAVIYEDNRRPSGVP
Lambda.txt      FLSASVGDRVTITCRASQ-GISSLAWYQQKPGKAPKLLIYAASTLQSGVP
                 :*.* *. :**:* .*: . :.: ****:**.**. :**  .   ****

Kappa.txt       DRISGTISSNSASLTISDLKTEDEAEYYCQA--
Lambda.txt      SRFSGSGSGTEFTLTISSLQPEDFATYYCQQLN
                .*:**: *... :****.*:.** * ****
```

Figure 4.15: Alignments generated by conventional algorithms for comparison with FIRE and the new BLOSUM-FIRE. Data sets used are the Lambda and Kappa light chain antibody variable regions.

T-COFFEE alignment

```
Lambda          -VSGSPGKTITISCSGSDIATTNIQWYQQRPGSAPTAVIYEDNRRPSGVP
Kappa           FLSASVGDRVTITCRASQ-GISSLAWYQQKPGKAPKLLIYAASTLQSGVP
                 :*.* *. :**:* .*: . :.: ****:**.**. :**  .   ****

Lambda          DRISGTISSNSASLTISDLKTEDEAEYYCQA--
Kappa           SRFSGSGSGTEFTLTISSLQPEDFATYYCQQLN
                .*:**: *... :****.*:.** * ****
```

MUSCLE alignment

```
Lambda          -VSGSPGKTITISCSGSDIATTNIQWYQQRPGSAPTAVIYEDNRRPSGVPDRISGTISSN
Kappa           FLSASVGDRVTITCRASQ-GISSLAWYQQKPGKAPKLLIYAASTLQSGVPSRFSGSGSGT
                 :*.* *. :**:* .*: . :.: ****.**.**. :**  .   ****.*:**: *..

Lambda          SASLTISDLKTEDEAEYYCQA--
Kappa           EFTLTISSLQPEDFATYYCQQLN
                . :****.*:.** * ****
```

Figure 4.16: Comparison of alignments generated by FIRE and other algorithms continued.

The algorithms were then compared against each other to align the analogous: κ and λ antibody variable regions domains used to evaluate the FIRE algorithm in Section 3.3. The alignments for MAFFT, CLUSTAL Omega, MUSCLE and T- COFFEE are provided in Figure 4.15 and continued on Figure 4.16. Additionally, alignments produced by the a BLOSUM62 matrix based algorithm are provided for comparison to demonstrate the difference between the new algorithm and a BLOSUM based algorithm.

The modifications and improvements to the algorithm were evaluated by comparing the performance measured by residue identity score using datasets described in Table 3.2. Table 4.4 shows the performance of the BLOSUM-FIRE algorithm compared against widely used algorithms. The results reveal that BLOSUM-FIRE and MAFFT had overall best performances compared to the other algorithms.

The quality of alignments produced by BLOSUM-FIRE were compared to those of the FIRE. The experiments which led to these results are described in Section 3.7.

The similarity of the ω was plotted against codon position for 3 pairs of data sets. These plots are presented in Figures 4.17 and demonstrate the differences between the alignments generated by the BLOSUM-FIRE algorithm and FIRE.

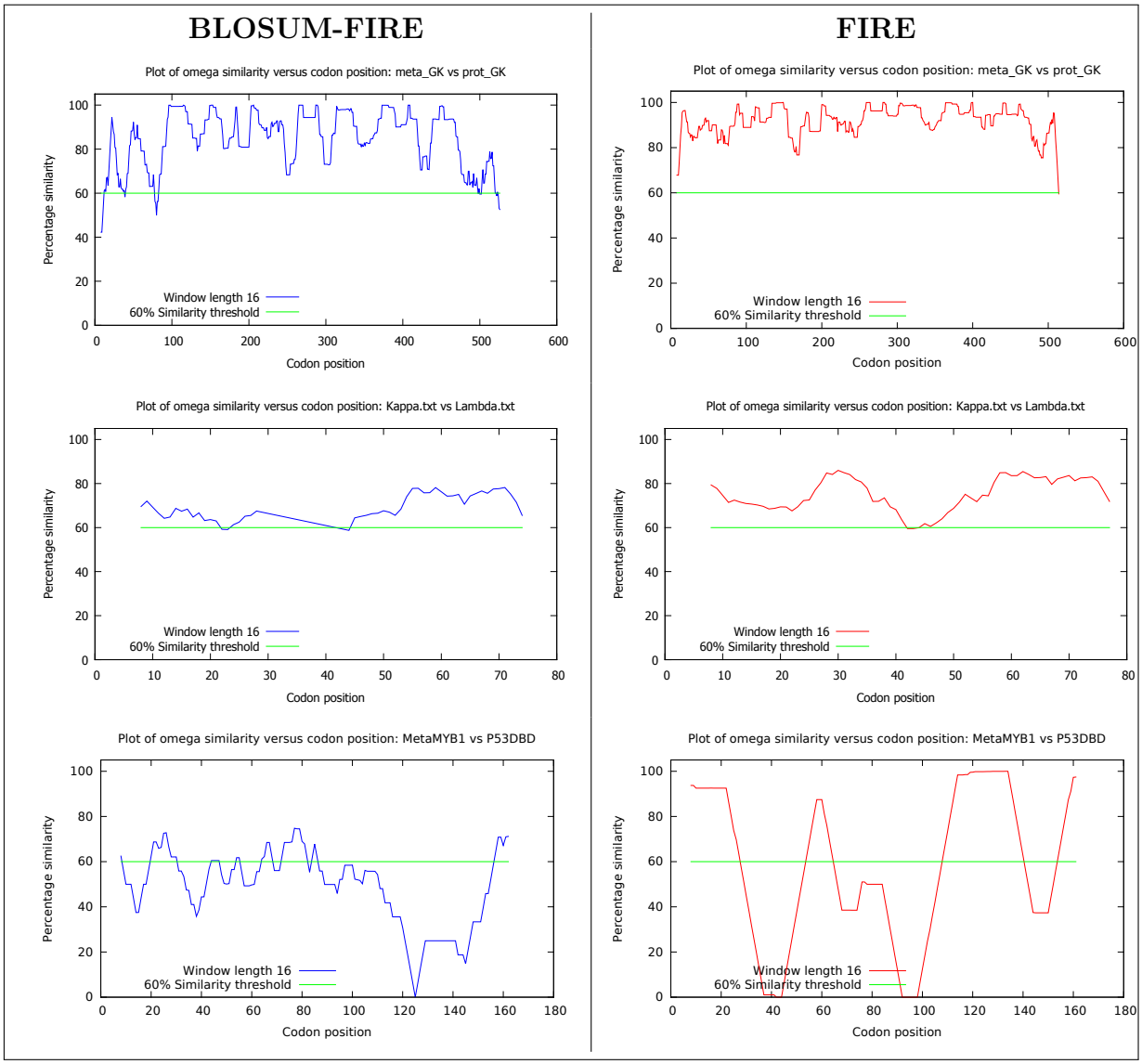


Figure 4.17: Similarity plots of MLEs at codon sites for FIRE and BLOSUM-FIRE representing the new algorithm, presented as a percent similarity between the two ω values. The datasets: meta_GK is conserved metazoan Glycerol Kinase, prot_GK is protozoan Glycerol Kinase; kappa and lambda are Kappa and Lambda antibody regions, p53DBD is P53 DNA binding domain and meta_myb1 is metazoan MYB1.

4.7 Determining the optimal gap penalties using the variable gap penalty scheme

The empirical approach described in Section 3.6.1 was used to determine optimal gap penalties for the BLOSUM-FIRE algorithm.

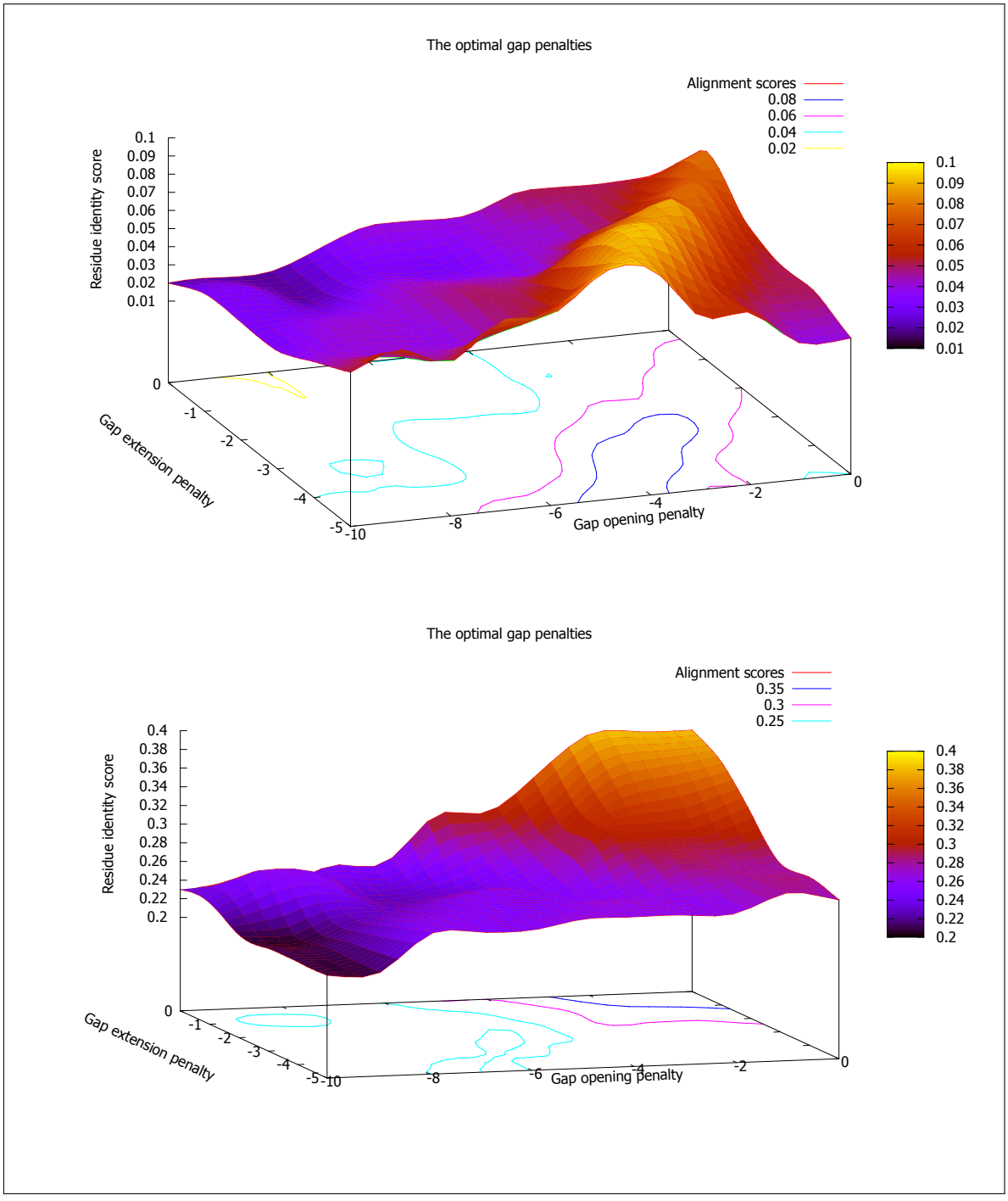


Figure 4.18: The optimal gap penalties. Plots were produced using the median of the 5 lowest scores (above) and the median of the highest 5 scores (below). Scores were obtained from 6930 alignments from iterating through the gap parameters using an affine gap penalty scheme. The orange sections of the heat map indicate the region of the optimal penalty.

Results of the algorithm identity scores obtained from iterations of the gap open penalty from 0 to -11 and iteration of the gap extension penalty in the range [0,-6] were used to generate Figure 4.18. The datasets used in this study are described in Section 3.6.1. The plots show heat maps for the optimal gap open penalty and gap extension penalty for distantly related proteins.

4.7.1 The efficacy of pruning phylogenetic trees in the compilation of EvoDB

Phylogenetic trees were required for the CODEML program to calculate the ω profiles at codon sites using options and parameters provided in Section 3.2. However, the PFAM database provides the phylogenetic tree corresponding to the PFAM family. A computationally less expensive approach described in Section 3.8.5 to prune the existing phylogenetic trees instead of calculating new trees was adopted. To evaluate this adopted approach, trees produced by pruning and those generated from the nucleotide sequence data were compared. The FIGTREE version 1.4.0 program was used to draw the phylogenetic trees as described in Section 3.8.5. Figure 4.19 shows the phylogenetic tree before experiment of evaluating the pruning algorithm.

Phylogenetic trees generated by pruning using the Prune function on Biopython and FastTree program were compared. These methods were compared to identify any difference in the quality of trees produced. The trees files produced by these two approaches are presented in Figure 4.20 and the corresponding phylogenetic tree diagrams are also provided in Figure 4.21 for comparison.

4.8 The development of the EvoDB database

A database of ω MLEs values was required for the inference of domain functions using the BLOSUM-FIRE algorithm. Section 3.8 provides the implementation details of the Evolutionary Rates atabase (EvoDB). The infrastructure and computational resources

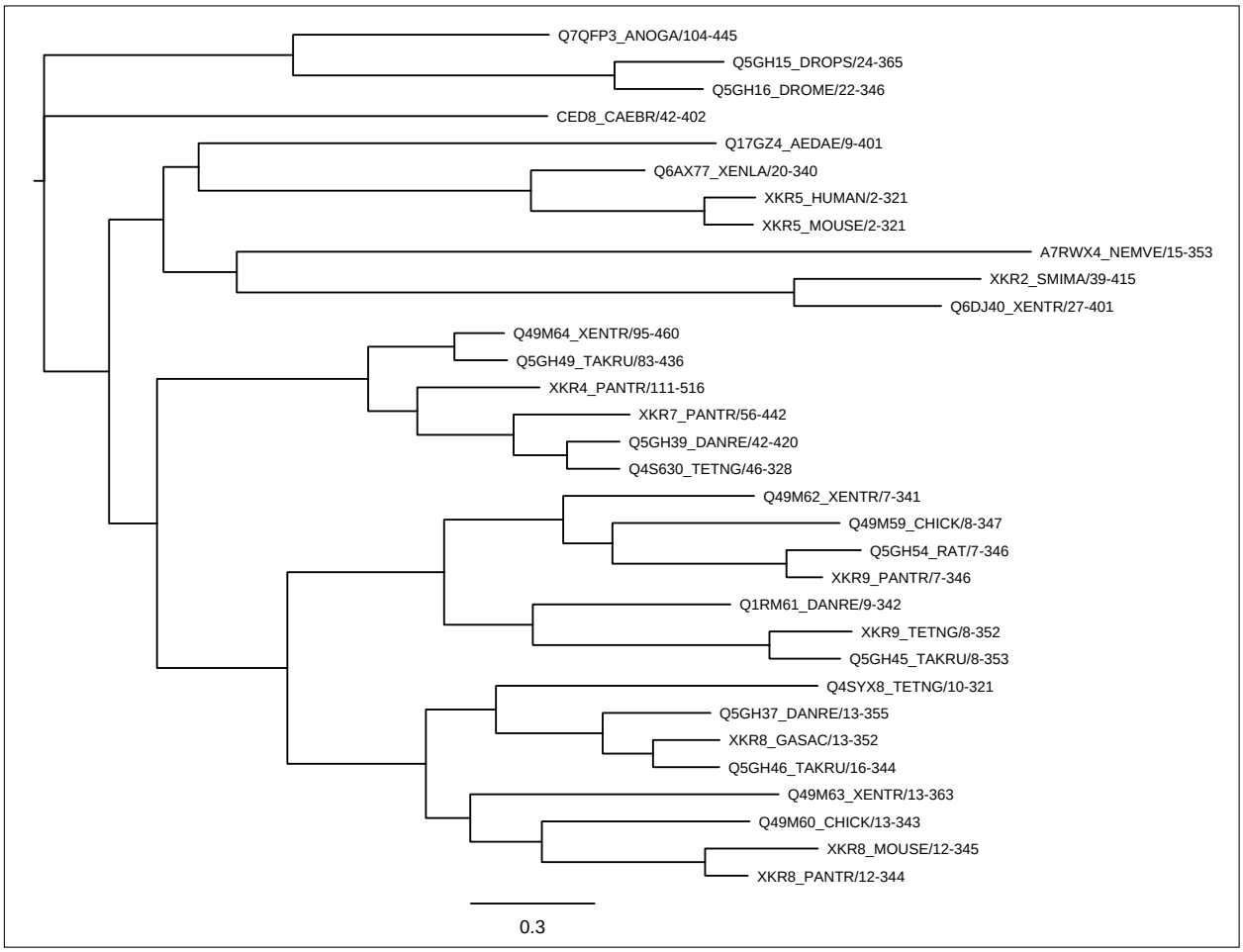


Figure 4.19: Phylogenetic tree showing the sequences as nodes found in the PF09815 family. The members of the family share domains related to the XK proteins, that are involved transport of various proteins.

Pruned tree

```
((XKR8_MOUSE/12-345:0.19465,XKR8_PANTR/12-344:0.13364)1.00:0.64295,(CED8_CAEBR/42-402:0.71356,A7RWX4_NEMVE/15-353:0.95909)0.64:0.03971):1.00000;
```

Calculated tree

```
(XKR8_MOUSE/12-345:0.22292,XKR8_PANTR/12-344:0.12980,(A7RWX4_NEMVE/15-353:1.77557,CED8_CAEBR/42-402:0.84380)0.999:1.18655);
```

Figure 4.20: Phylogenetic tree files produced by pruning (pruned tree) and calculated from the alignment using the FastTree algorithm (calculated tree).

of the Wits Core Cluster (ZA-WITS-CORE) were used to implement the EvoDB. A total of 13672 families were found in the PFAM-A seed alignments database version 27.0 database with a total of 678132 UNIPROT (Consortium et al., 2013) protein

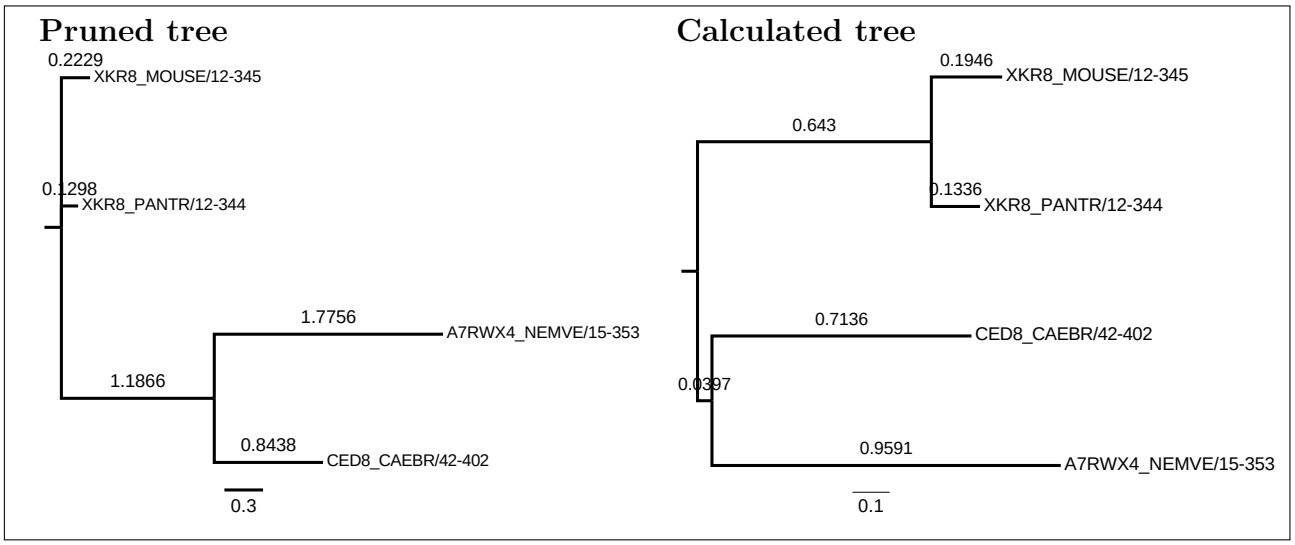


Figure 4.21: Comparison of phylogenetic trees produced by the pruning algorithm from the Phylo class in Biopython and calculated using the FASTREE program. Tree diagrams were generated using the FIGTREE program.

sequences combined. A total of 14 days were required to complete the first phase of the compilation of EvoDB. However, some of the PFAM families were not processed or compiled on EvoDB as the computation of the BEB ω MLEs using the CODEML program (Section 3.2) exceeded the cluster wall time requested on the MAUI/PBS scheduling system of 240 CPU hours and were therefore terminated before completion. Those families that were not compiled or had not completed the ω MLE calculations were resubmitted in a second phase of compiling EvoDB and allocated more time, a fraction of these were able to complete.

4.8.1 EvoDB coverage of the PFAM-A protein domain database

EvoDB was compiled (Section 3.8) for query by the BLOSUM-FIRE algorithm to infer domain functions of novel proteins. The EvoDB coverage of the PFAM-A seed alignments was analysed and the results are presented here. A total of 520885 sequences from the total 678132 were retrieved from the SWISS-PROT, TrEMBL and GenBank databases completely. Therefore, 157247 protein files were not processed due to miss-annotations errors and other program errors which led to failures in retrieving sequences from SWISS-PROT and GenBank databases. Table 4.5 shows the numbers

and percentages of PFAM families whose ω profiles were calculated and compiled on the EvoDB database.

Table 4.5: The number of compiled families and their percentages.

PFAM Families	Numbers	Percentage
Compiled with ω profiles	13 457	98.4%
Incomplete families without ω profiles	215	1.6%
Total	13 672	

A fraction of the PFAM families (1.6%) were not compiled due to miss-annotation errors and other errors in the pipeline, the source of these errors was due one or more programs in the pipeline failing as provided in Table 4.7. Most of these profiles were not compiled due to the large number of sequences their families which caused the jobs to exceed the allocated computation time on the cluster.

Table 4.6: Analysis of the compilation of EvoDB. The table shows the numbers and percentages of protein sequences that could not be retrieved from either SWISSPROT, TrEMBL and GenBank databases.

PFAM Protein Sequences	*Numbers	Percentage
Found	520885	77.8%
Not Found	157247	22.2%
Total	678132	

*These numbers include those sequences that failed validation by the TRANALIGN program or coding sequences that could not be extracted by the EXTRACTFEAT utility.

Table 4.7: The numbers of protein sequences that failed retrieval[1].

Utility	Number of protein sequences failing	Percentage of total fails
ENTRET (SWISSPROT)	9186	5.8%
ENTRET (GENBANK)	2368	1.5%
EXTRACTFEAT	56 469	35.91%
TRANALIGN	87 340	55.54%
Cross references issues	1884	1.20%
Total	157 247	

[1]The list is separated according to the EMBOSS utility that failed. The highest percentage of errors were related to validation issues where the nucleotide sequences retrieved did not match the protein sequences.

The compilation of EvoDB was analysed and the analysis of sequence failing retrieval is provided in Table 4.6 and by separated by utility failing in Table 4.7. These results reveal that the highest percentage of sequences failed the TRANALIGN algorithm validation step, where the nucleotide sequences are validated against the protein sequences. This indicates that the wrong nucleotide sequences were retrieved from the GenBank database.

4.8.2 EvoDB results for HBx query

The evolutionary rate profiles of the HBx protein were compared to the profiles in EvoDB (Section 3.9). The results of the top five statistically significant results from querying the EvoDB database using the HBx are displayed in Table 4.8.

Table 4.8: The top 5 statistically significant results of HBx aligned against EvoDB PFAM profiles. The results presented here are sorted according to the FIRE score.

*PFAM Id	[1]FIRE score	ID	†ω sim	Residues matched	*p*-value
PF00739.14	0.62	76	0.49	113	0.0
PF05407.7	0.49	20	0.77	33	7.62×10^{-05}
PF10895.3	0.48	18	0.77	27	2.28×10^{-04}
PF03866.8	0.47	18	0.76	29	3.81×10^{-04}
PF05417.6	0.46	25	0.67	41	6.85×10^{-04}

*PFAM Id is the accession number of the PFAM family, [1]FIRE score is the BLOSUM-FIRE algorithm score, PID is the percentage identity score of the aligned sequences †ω sim is the similarity of the ω values and represents what used to be the FIRE score and residues matched is the number of identical amino acid matched from the two sequences.

The results show that the FIRE algorithm was able to identify the HBx family, PF00739.14, as the most statistically significant. Comparison of the old FIRE score and BLOSUM-FIRE score revealed the superiority of the evolutionary rate and BLOSUM matrix coupled approach over the two approaches individually.

The inference of homology requires assessment of the statistical significance of an alignment to identify those real alignments from those due to chance as discussed in Section 2.7. A $p-$value statistical framework was implemented to assess the signifi-

cance of HBx results on the EvoDB database. The implementation of this statistical framework using a shuffled HBx ω MLE profile is provided in Section 3.9.1.

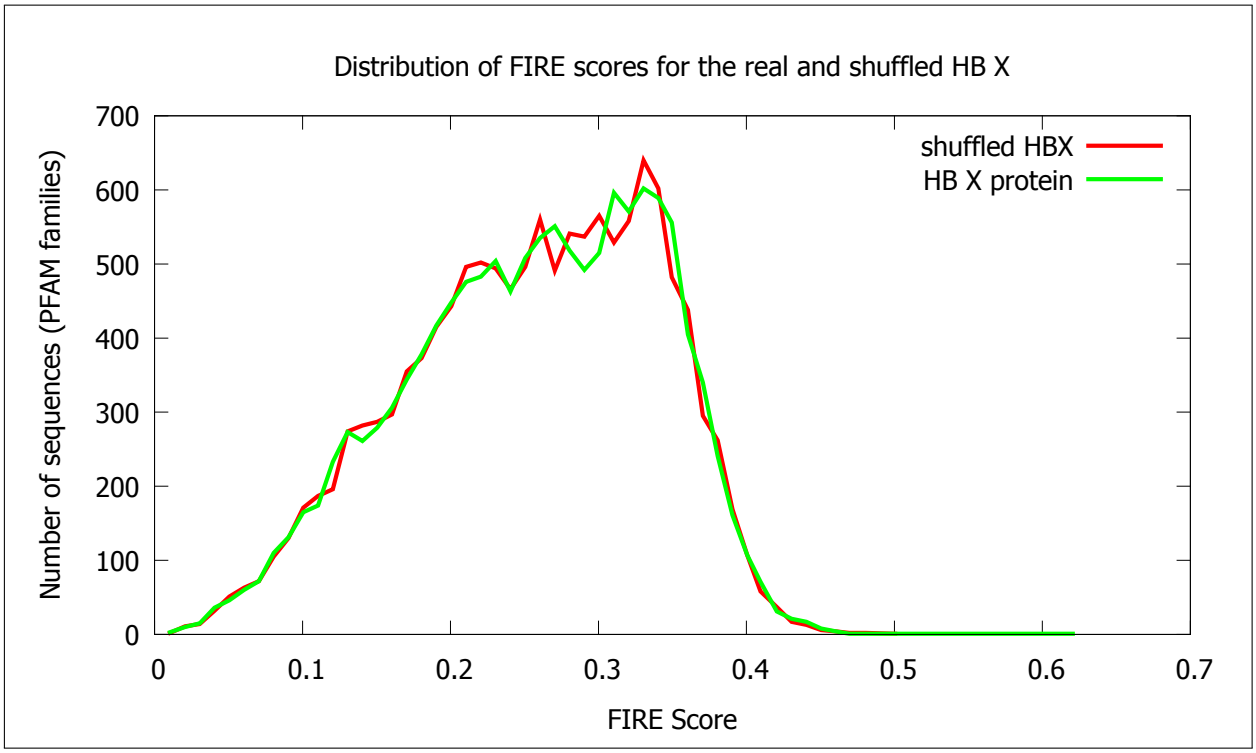


Figure 4.22: The distibution of FIRE scores for a shuffled HBx and the actual HBx. The approximate normal distrubution of HBx and shuffled HBx alignment scores were used to calculate the statistical framework.

4.9 The inferred functions of the enigmatic Hepatitis B Virus X protein

The descriptions of the top scoring families in EvoDB found by the FIRE algorithm were retrieved and presented in Table 4.9 and their scores are shown in Table 4.8 using the statistical framework described in Section 3.9.1. Using the normal distribution of the alignment scores, a significance level of 0.01 was chosen. The plot shown in Figure 4.22 shows the statistical distribution of the scores for the actual HBx query and a shuffled HBx query.

The top two scoring alignments are presented, first in Figure 4.23 showing the Trans-activation protein X family with HBx and second in Figure 4.24 showing the

Table 4.9: Description of the top hits for the HBx protein on EvoDB.

PFAM Id	*Description of family
PF00739.14	Trans-activation protein X family. This protein is found in hepadnaviruses where it is indispensable for replication.
PF05407.7	Rubella virus endopeptidase family. Corresponds to Merops family C27. Required for processing of the rubella virus replication protein.
PF03866.8	Hydrophobic abundant protein (HAP) family. Expression of HAP is thought to be developmentally regulated and possibly involved in spherule cell wall formation.
PF08443.6	RimK-like ATP-grasp domain family. This ATP-grasp domain is found in the ribosomal S6 modification enzyme RimK.
PF10895.3	Protein of unknown function (DUF2715). This family of proteins with unknown function appears to be restricted to Treponema pallidum.

*Descriptions were obtained from the EvoDB PFAM files.These have been presented in the order of statistical significance provided in Table 4.8. The PFAM Id is the PFAM version 27.0 database accession identifier for the family.

```
BLOSUM-FIRE Score : 0.62
Percent Identity score : 0.76
Residue Matches : 113/153
dN/dS similarity score(old FIRE) :0.49

x_protein.fire   MAARVCCQLDPARDVLCLRPVGAESRGRPVSGPFGTLPSPS..SAVP.AH
PF00739.14       MAARVCCQLDPARDVLCLRPVGAESRGRPLPGPLG.ALPPASPSAVPTDH

x_protein.fire   GAHLSLRGLPVCAFSSAGPCALRFTSARRM.TTVNAHQLPKVLYKRTLGL
PF00739.14       GAHLSLRGLPVCAFSPAGPCALRFTSARRMETTVNL...SKVLHKRTLG.

x_protein.fire   AMSTTDLEAYFKDCLFKDWEELGEEIRLMIFVLGGCRHKLVCSPAPCNFF
PF00739.14       .MSTTDLEAYFKDCVFTEWEELGEEMRLMIFVLGGCRHK...L.V.....

x_protein.fire   TSA
PF00739.14       ...
```

Figure 4.23: FIRE alignment for the top scoring alignment of the HBx protein, PF00739.14 is the Trans-activation protein X family of domains where HBx is found.

HBx aligned with the rubella virus endopeptidase family.

A plot of the similarity of the ω plotted against codon position is presented in Figure 4.25. The plot reveals that the ω MLEs distribution for the two proteins have high similarity and according to the FIRE principle they are under very similar selective pressures. On the other hand, the results (Table 4.8) show similar scores for the other

```
BLOSUM-FIRE Score : 0.49
Percent Identity score : 0.20
Residue Matches : 33/169
dN/dS similarity score(old FIRE) :0.77

x_protein.fire    MAARVC...CQL..DP..ARDVLCLR..PVGAESRGRPVSGPFGTLPSPS
PF05407.7         WRCRGWQGMPQVRCTPSNAHAALCRTGVPPRVSTRGGELDPNTCWLRAAA

x_protein.fire    .SA..VPAHGAHLSLRGLPVCAFSSAGPCAL...RFTS.ARRMTTVNAHQ
PF05407.7         NVAQVARACGAYTS.AGCPKCAYGRALSEARTHEDFAALSQRWSASHADA

x_protein.fire    LPKVLYKRTLGLAMSTTDL..EAYFKDCLFKDWEELGEEIRLMIFVLG..
PF05407.7         SPDGT.GDPLDPLMETVGCACSRVWVGS.EQEAPPDHLLVSLHRAPNGPW

x_protein.fire    GCRHKLVCSPAPCNFFTSA
PF05407.7         GVVLEVRARPEGGNPTGHF
```

Figure 4.24: A statistically significant alignment for HBx protein and PFAM family PF05407.7 corresponding to the rubella virus endopeptidase family.

PFAM families, these results illustrate the limitations of the FIRE algorithm when aligning highly conserved data sets.

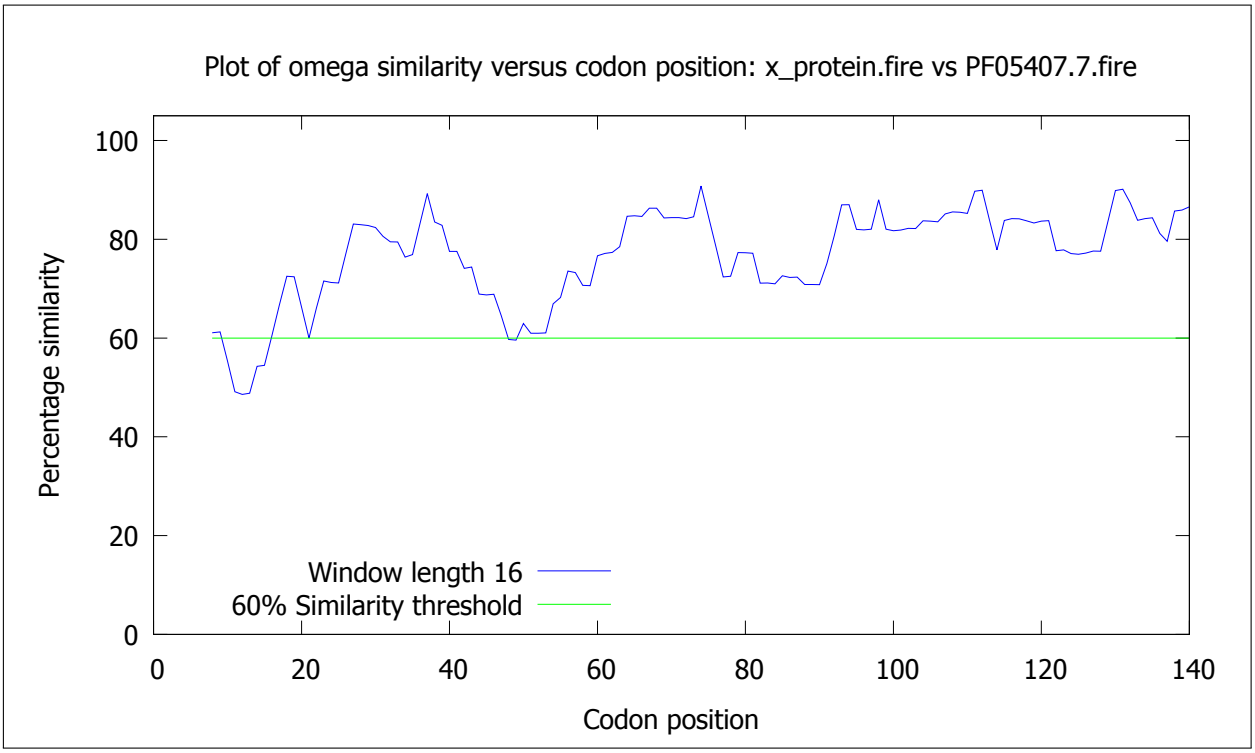
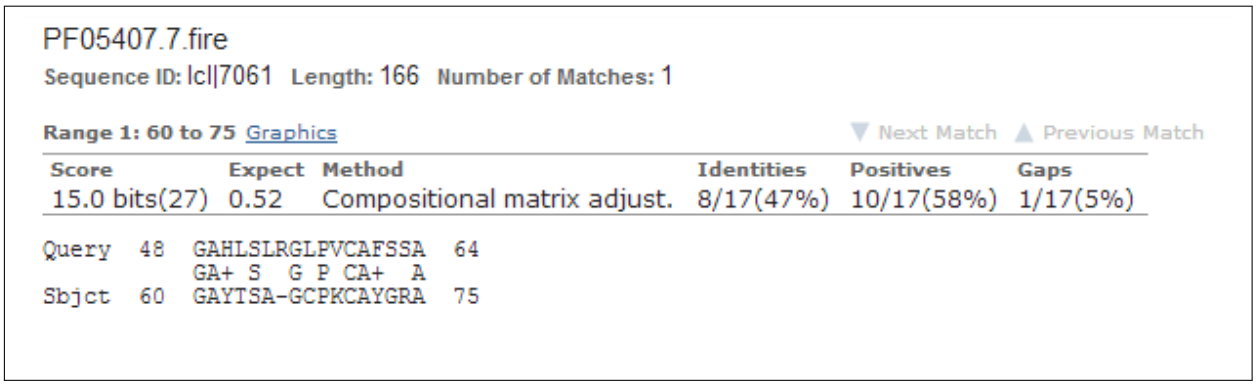


Figure 4.25: Plot of ω similarity against codon position for the HBx and rubella endopeptidase protein family. The green line indicates the 60% similary threshold, the HBx and rubella endopeptidase protein ω distributions are similar.

The top 10 statistical significant results (Table 4.8) found by BLOSUM-FIRE on

EvoDB were aligned to HBx using the BLAST algorithm webserver at www.ncbi.nlm.nih.gov/blast, however, only the PF05407.7 and PF00739.14 families were found to be statistically significant by the BLAST algorithm. Figure 4.26 shows the region of similarity found to be statistically significant by the BLAST algorithm.


PF05407.7.fire

Sequence ID: lcl|7061 Length: 166 Number of Matches: 1

Range 1: 60 to 75 Graphics ▼ Next Match ▲ Previous Match

Score	Expect	Method	Identities	Positives	Gaps
15.0 bits(27)	0.52	Compositional matrix adjust.	8/17(47%)	10/17(58%)	1/17(5%)

```
Query  48  GAHLSLRGLPVCAFSSA  64
           GA+ S  G P CA+  A
Sbjct  60  GAYISA-GCPKCAYGRA  75
```

Figure 4.26: BLAST alignment for the HBx and rubella virus endopeptidase sequence. This was the only statistically significant alignment found by the BLAST algorithm.

Analysis of these results revealed that the alignment had a bit score of 15 and an E-value of 0.52 with 8 identical matches for the region identified by the BLAST algorithm. The alignment produced by the FIRE algorithm was analysed on the MEME webserver at http://meme.nbcr.net/. Using the default parameter for the MEME algorithm, the algorithm was able to identify similar motifs in the two sequences as provided in Figure 4.27.

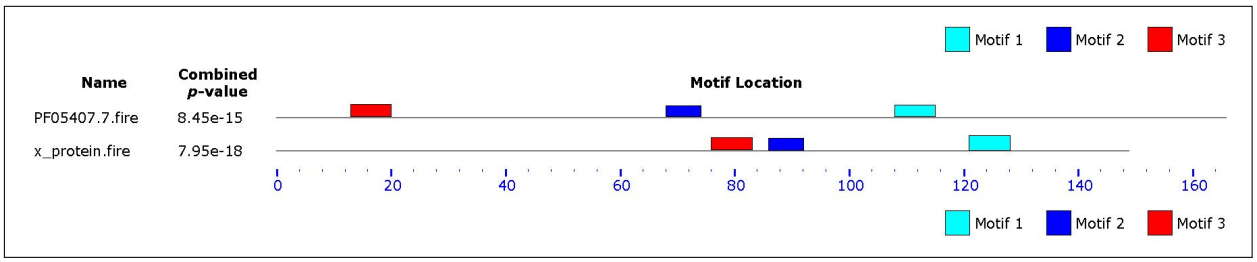


Figure 4.27: Motifs found in the two proteins. The MEME algorithm found 3 different motifs in the two proteins.

Chapter 5

Discussion

The FIRE algorithm was conceptualised to address the challenge of low sequence similarity in alignments. However, challenges arose when the algorithm produced false positives in certain datasets. The main objective of this research was to produce a new tool that uses an evolutionary rate based approach with increased sensitivity in the scoring function and increased specificity in alignments produced. The overarching aim of this study and the concept paper of Durand et al. (2010) was to address the general challenge of low sequence similarity in what has been called the "twilight zone" of sequence alignment. However, for any sequence alignment tool to be useful it has to satisfy the basic requirements of the user and the main requirement is that it must have a significantly low false positive rate. This study addresses the problem of false positives in the FIRE algorithm by coupling the FIRE approach with an substitution matrix approach, BLOSUM, in a dynamic scoring function.

The chapter begins by discussing the efficacy of the evolutionary rate (FIRE) approach to address the challenge of comparing amino acids sequences. The BLOSUM62 substitution matrix was introduced to address the problem of false positive results for cases where sequence conservation was high. The utility of coupling the BLOSUM approach with the FIRE approach was evaluated. The results of the evaluation of these two approaches which resulted in the conceptualisation of the new algorithm

were assessed. The results of the quality performance of the BLOSUM-FIRE algorithm in comparison with widely currently used sequence analysis tools will also be discussed in this chapter. The variable gap penalty scheme that was adopted and the empirical approach of determining optimal gap penalties used in this study will also be assessed. Furthermore, the efficacy of the compilation of the EvoDB database for inferring domain functions is also evaluated. The alignment results of the Hepatitis B Virus X protein on the EvoDB database are evaluated and the statistical framework used to determine the statistical significance of the top alignments is reviewed. The FIRE algorithm is evaluated and the results of the query on EvoDB presented in the previous chapter and how they address the overall aim of inferring function when sequence similarity is low. Finally, in the last section to the chapter the study reviews how the FIRE algorithm can be improved and provide suggestions for future work.

5.1 Addressing the challenge of false positives in the FIRE algorithm

This section discusses the results presented in Sections 4.1, 4.2 and 4.3, where the FIRE algorithm was evaluated and limitations resulting in false positives were identified. These false positives were characterized by good FIRE scores for poor alignments Durand et al. (2010), particularly for highly conserved datasets, which meant that the hypothesis of using the evolutionary rate as an alignment metric has limitations when sequences are highly conserved. Furthermore, the results of the evaluation of the current algorithm scoring function and their influence on the conceptualisation and formulation of the new algorithm are discussed.

The approach taken in this research was to identify limitations in the algorithm conceptualisation and produce a more specific algorithm whilst still based on the concepts and hypothesis proposed in the initial study. Whilst other approaches such as adjusting the gap penalty parameters similar to the modifications made to BLAST in Gapped

BLAST and PSI-BLAST (Altschul et al., 1997) or simply modifying the calculation of scores can be suggested. Such modifications would not address the challenge of poor alignments as there would be resultant loss of specificity in the algorithm. Additionally, it was an important consideration at each point of the study to ensure that the score and alignments had biological relevance that could be used to test hypothesis of homology between sequences (Kuzniar et al., 2008). Therefore, the challenge addressed by this study was to reduce the rate of false positives by increasing the specificity of the scoring function and the sensitivity of alignments produced. To achieve this, amino acid residue identities could no longer be ignored. This resulted in the integration of scoring amino acid residues into the FIRE algorithm as proposed using guidelines provided in Section 3.4.2 to address the challenges that arise in cases of high sequence conservation.

The FIRE algorithm was able to align sequences using the evolutionary rate as a similarity metric whilst ignoring the amino acids residue identity (Table 4.1), thus corroborating the findings and supporting the hypothesis of Durand et al. (2010). The canonical example of a false positive result produced by the FIRE algorithm is shown on Figure 4.2 generated using simulated highly conserved ω MLEs data in the range [0.050,0.053]. Equally important, are the results in Figure 4.3 which demonstrated that residue based approaches were superior to the FIRE approach for highly conserved datasets. Therefore, the results of the evaluation of the algorithm support the findings of the concept paper of Durand et al. (2010) that the algorithm can be used to infer domain function. On the other hand the results also supported the finding of false positives in highly conserved datasets.

Statistical analysis of the distribution of ω values in those data files (Table 4.2) producing false positives is and the alignment scores (Table 4.1). In this analysis comparison of median values and alignment scores revealed that there was a negative

correlation between the alignment score and magnitude of median difference in ω values. This correlation was found to be attributed to the fact that the FIRE algorithm compares the ω MLEs values as a similarity metric. This was confirmed by the results of the statistical analysis of the distribution of the ω in the datasets which revealed that the false positives produced by FIRE were due to the low specificity of the algorithm scoring function.

Section 4.3 presents the outcome of the comparison of the FIRE and BLOSUM approaches and the results displayed in Table 4.3 demonstrate the poor sensitivity of the FIRE algorithm compared to a BLOSUM based approach. One of the limitations of the FIRE algorithm identified in the concept paper and in the results provided in previous chapter was the reduced specificity when the sequence conservation is high. When the conservation is high and the variance of ω values is relatively low, it results in high similarity scores all throughout the alignment. Furthermore, analysis of some of the alignments showed a high FIRE score, however, when the alignment was inspected there were few residue matches. This problem was a result of the scoring function based on Equation (3.2) of the FIRE algorithm described in Section 3.1 which explicitly uses the similarity of the ω values as an indicator of shared selective pressures and similar functions according to the hypothesis of the algorithm. Evaluation of alignment produced by the FIRE algorithm provided conclusive evidence that the homogeneity of the ω values at codon sites reduces the specificity of the algorithm; this is seen in highly conserved sequences where there was a correlation between low variance in the ω values and false positives were produced.

It is however important to note that the alignments produced by the FIRE algorithm are biologically more likely than the alignments produced by a BLOSUM matrix based approach. This is because biological sequences are more likely to have long few gaps than numerous small gaps (Gotoh, 1990). The dynamic global alignment algorithm of

Needleman-Wunsch underlying the sequence alignment tool has also been identified to play a role in the insertion of gaps in the alignment.

5.2 The utility of BLOSUM substitution matrices in sequence alignment

To address the challenge of false positives the BLOSUM family of matrices was identified for coupling with the evolutionary rate approach to produce a more sensitive algorithm. The use of substitution matrices is a conventional approach (Altschul, 1991) and numerous of these have been provided for scoring aligned residues as described and discussed in Section 2.4. BLOSUM-FIRE algorithm joins the plethora of widely used sequence alignment algorithms that use substitution matrices such as MAFFT (Katoh and Toh, 2008), CLUSTAL W (Thompson et al., 1994), T-COFFEE (Di Tommaso et al., 2011) and BLAST (Altschul et al., 1990).

The BLOSUM matrices, specifically, the BLOSUM62 matrix was chosen as the amino acid scoring matrix for this study as it has been found to be the most effective of the BLOSUM matrices (Henikoff and Henikoff, 1993; Katoh and Toh, 2008). On the other hand, the BLOSUM-FIRE algorithm uses the Needleman-Wunsch algorithm and in some literature it is suggested to use the PAM matrices for global alignments and BLOSUM matrices for local Smith-Waterman alignments. Given that some of this work was hypothesis based the choice was made to use matrices derived from real data sets such as BLOSUM to reduce systematic errors instead of using extrapolated matrices such as the PAM matrices. Studies have also shown that the BLOSUM matrices are more effective at detecting distantly related proteins than the PAM matrices (Katoh and Toh, 2008). Aligning distantly related proteins is a challenge that BLOSUM-FIRE was conceptualised to address. This decision was also influenced by the plethora of literature citing the efficacy of the BLOSUM matrix and of particular the BLOSUM62 (Altschul et al., 1997; Katoh and Toh, 2008; Di Tommaso et al., 2011; Sievers et al.,

2014).

5.3 Comparison of FIRE and a BLOSUM based approaches

To effectively combine the FIRE and BLOSUM approaches they had to be independently evaluated as described in Section 3.4.1 and the results are presented in Section 4.3. Using the number of residue matched as a performance measure the BLOSUM based algorithm was found to have performed better than the FIRE algorithm in all the data sets as provided in the alignment scores shown in Table 4.3. On the other hand, the BLOSUM matrix based algorithm was not affected by conservation at sequence level and produced more accurate alignments. However, analysis of the results and datasets show that the FIRE algorithm was able to identify those points of positive selection inferred from the ω value to insert gaps and therefore produce more biologically significant alignments.

The results of the comparison of the FIRE and BLOSUM based approaches also confirmed the results in the initial study that the FIRE algorithm performs well in sequences of varied conservation and performs poorly in highly conserved data sets. Conversely, the BLOSUM matrix scoring approach does not identify areas of varied conservation and therefore missed those codon sites of positives selection which may provide valuable information to a researcher. At this point it is important to point out that it is not expected for the BLOSUM-FIRE algorithm to outperform conventional scoring matrix based approaches when using residues matches as a performance measure but expect the algorithm will produce the most biologically relevant alignment.

Comparison of the quality of alignments using widely used amino acid based approaches in this study support the literature that asserts that scoring matrix based methods are superior to other approaches that uses information related to and not

the amino acids themselves. Finally, the results of this study extend the evidence that amino acid substitution matrix based methods are likely to be used as scoring matrices for sequence alignment algorithms in the near future (Yamada and Tomii, 2013). However, recent work has provided accurate alignment methods that do not use substitution matrices (Angermüller et al., 2012).

5.4 The efficacy of a scaled BLOSUM score

One of the important aspects of BLOSUM-FIRE is the scaling of BLOSUM62 score using the selective pressure at the amino acid sites. In Section 3.5.1 the scaling of the BLOSUM score using Equation (3.4) is provided. This scaling of BLOSUM scores adjusts the BLOSUM scores according to the selective pressure at the amino acid site. Equation (3.4) uses the selective pressure inferred from the ω values at codon sites to scale the BLOSUM score. The rationale is that the choice of which BLOSUM matrix to use, is usually based on some *a priori* knowledge regarding the similarity of sequences, however, at the amino acid site level the conservation of residues varies. Therefore, this adjustment using the selective pressure approximation means that the same amino acids under different selective pressures will have different scores. The basis for this is that highly conserved residues have ω values approaching zero and those highly evolving sites have ω values approaching and greater than one, interpreting ω values is discussed in Section 2.11.

Equation (3.6) whose conceptualisation is described in Section 3.5.1 was proposed to approximate the conservation of residues at amino acid levels and is used as a proxy for the selective pressure at codon sites. While better approximations can be conceived by using advanced methods such as machine learning or hidden Markov models to generate a scoring function, the improved sensitivity in the algorithm will be negligible as the context of the alignment must be taken into account. The role played by other factors such as the dynamic algorithm, be it Needleman-Wunsch (Needleman

and Wunsch, 1970) or Smith–Waterman (Smith and Waterman, 1981), gap penalties (Gotoh, 1990), divergence in sequence length, similarity of the sequences aligned and substitution matrix (Henikoff and Henikoff, 1993) chosen must not be ignored.

Analysis of Equation (3.6) for using the selective pressure to scale the BLOSUM score reveals that it scales down positive BLOSUM scores (match scores) and upscales negative BLOSUM scores (mismatch scores). This was an implementation choice and the rationale was not to penalise negative BLOSUM scores especially in codon sites of positive selection where the identity of the amino acid is variable. Therefore, in the BLOSUM-FIRE algorithm identical poorly conserved amino acids under positive selection amino acid do not score the same as identical highly conserved amino acids.

5.5 The efficacy of a proportionality constant

A proportionality constant (K) in Equation (3.11) allows the FIRE and BLOSUM matrix approaches to be comparable. The proportionality constant (K) is a scaling factor, it is the mean of identical amino acid match scores for the BLOSUM62 matrix. An assumption that states that a match in ω values is analogous to a match in residues in a BLOSUM substitution matrix was adopted. Therefore, the mean of the identical match scores of the amino acid matches was scaled to the similarity between two ω using sim_{ab} and the proportinality constant as presented in Equation (3.9).

Finally, to make the FIRE scores comparable to BLOSUM scores the proportionality constant of the BLOSUM62 matrix was used in Equation (3.11). This was only implemented for the BLOSUM62 matrix and can be extended to the rest of the family of the BLOSUM matrices, however, the efficacy of using a different substitution matrix from the BLOSUM family was not investigated.

5.6 The new scoring function

To address the main objective of improving the performance of the algorithm in terms of quality of alignments, a new scoring function was proposed and this resulted in the implementation of a new algorithm. The new scoring function in the form of Equation (3.10) is dynamic using selective pressure to balance: using the FIRE principle for sequences of varied conservation at codon sites and using BLOSUM principle for highly conserved codon sites. The concept behind the design of the BLOSUM-FIRE algorithm was to use the BLOSUM matrices for scoring residues in which the FIRE algorithm was producing false positives in highly conserved datasets. The balance between the BLOSUM and FIRE principle depends on the conservation at amino acid site level determined by Equation (3.6), this results in two identical amino acid pairs being scored differently based on the level of conservation. In addition, the scoring function is monotonic and the move from highly conserved sequences to sequences of poor conservation is continuous. Equally important, the coupling of the an evolutionary rate base approach and the BLOSUM approach resulted in performances comparable to conventional algorithms as presented by the results in Section 4.6 and highlighted in the next section.

5.7 The performance of the new BLOSUM-FIRE algorithm

The BLOSUM-FIRE algorithm and the FIRE algorithm including four widely used sequence alignment algorithms were compared for their performance in terms of quality of alignments in Section 3.7. These alignments are displayed in Figure 4.15 and the corresponding scores in Table 4.4, demonstrate the improved quality of the BLOSUM-FIRE algorithm. These results also demonstrate the increased sensitivity in the alignments of the new BLOSUM-FIRE algorithm compared to the FIRE algorithm. Improvements in quality of alignments for the BLOSUM-FIRE algorithm were identified in highly

conserved sequences as provided in Section 4.6.

Analysis of the alignments from the different algorithms revealed that the BLOSUM-FIRE alignments are different when compared with other algorithms with different gap insertion sites depending on the level of conservation. The results also provided significant evidence of improvement in the BLOSUM-FIRE algorithm, specifically in terms of the numbers of correctly aligned residues. Comparison of the numbers of residue matches between the new BLOSUM-FIRE and FIRE revealed considerable improvements for almost datasets as displayed in Table 4.4. Finally, these comparison results also indicated that the performance of the algorithm is at par with conventional algorithms such as T-COFFEE, MAFFT and CLUSTAL W.

5.8 The variable gap penalty

Gap penalty parameter choice is a challenge in sequence alignment and has been the focus of several studies (Altschul and Erickson, 1986; Gotoh, 1990; Vingron and Waterman, 1994; Edgar, 2009; Carroll et al., 2006). Studies have demonstrated that modifications to the gap penalty schemes result in improved algorithm performance for example in Gapped BLAST (Altschul et al., 1997) and CLUSTAL W (Thompson et al., 1994) in which a position specific gap penalty was implemented. A variable gap affine penalty scheme discussed in Section 2.5 was implemented in the algorithm and the optimum gap penalties were empirically identified using distantly related sequences as described in Section 3.6. The gap penalty is scaled according to the level of conservation using guidelines discussed in Section 3.5.1. Those sites with elevated evolutionary rates ($\omega > 1$) have high probabilities for the insertions of gaps. The rationale for this was that those sites that have elevated ω values are highly variable substitution sites, interpreting the evolutionary rate is discussed in Section 2.11. To determine the optimal gap open and gap extension penalties distantly related evolutionary rate profiles of the PFAM-A database were used.

A variable gap penalty reduces the number of gaps in the alignment and ultimately produces more biologically informative alignments (Thompson et al., 1994). In this study a variable gap penalty was chosen as it is more sensitive to the conservation of the sequences. In cases of divergent sequence length the Needleman-Wunsch algorithm (Needleman and Wunsch, 1970) produces alignments with poor alignments with gaps, this is because the Needleman-Wunsch dynamic programming approach performs well for approximately equivalent sequence lengths this can be remedied by implementing a variable gap penalty or by modifying the algorithm so that it uses the Smith-Waterman approach which is effective for local alignments (Smith and Waterman, 1981). In addition, a terminal gap penalty can address the challenge of stretched sequences when there is high sequence length divergence (Thompson et al., 1994). The empirical analysis also revealed that FIRE performance decreases considerably when the test sequences have points of positive selection, the data has revealed that points of positive selection on one sequence corresponded with gaps on the other sequences.

Aligning sequences and the selection of gap penalties is a trade-off between the biological meaning of the alignment and computational implementation of the algorithm (Altschul and Erickson, 1986). While it is important that we obtain good alignments several fundamental questions arise: Do we want to get good alignments all the time? at what point do we consider alignments to be by chance or really based on some functional, structural, or evolutionary relationships? While frameworks for answering these questions have been provided (Section 2.5) and statistical frameworks have been proposed (Karlin and Altschul, 1990). These questions have become more important with the deluge of sequence data from high throughput projects.

In the determination of optimal gap penalties, it was also important to focus on those penalty parameters that maximised the detection of distantly related sequence similar to Edgar (2009) and this informed the selection and simulation of data sets. It

is important to bear in mind that there is a trade-off when increasing sensitivity for distantly related proteins and the reduction of specificity in closely related proteins. Fortunately, closely related protein have many matching sites and the match score and mismatch scores are sufficient in detecting those sites. In any case the scoring of the BLOSUM-FIRE algorithm is dynamic and therefore effective in detecting those differences. In such cases the divergence in sequence length will also have a correlation with the quality of the sequences aligned.

5.9 The evolutionary rates database (EvoDB)

EvoDB presents a tier of information untapped by current databases. A database of PFAM-A evolutionary rates was required to be queried by the BLOSUM-FIRE algorithm to infer domain functions of novel proteins as discussed in Section 3.8. The PANDIT (Protein and Associated Nucleotide Domains with Inferred Trees) database provides this information, however, the database has not been updated since version 17.0 of PFAM in 2005 (Whelan et al., 2006). Therefore, EvoDB contains the evolutionary rate profiles of the proteins families found in the PFAM protein domain database. Specifically, the EvoDB database contains a total of 520885 protein sequences found in 13672 families of which 13277 have had their evolutionary rates determined under the M2a model using the CODEML program found in the PAML suite of software (Yang, 2007). The phylogenetic trees and nucleotide sequence files have been computer curated and those sequences whose information could not be found were pruned from the tree files and nucleotide sequence files. Therefore, EvoDB represents a repository of information that would be helpful for researchers interested in evolutionary studies in protein domains. One of the challenges with compiling the EvoDB was that for 22% of the sequences, the nucleotide alignments could not be retrieved due to program errors and mis-annotation issues. The challenge of using the EvoDB is that all the PFAM-A ω MLEs profiles were determined using the same parameters for determining the Maximum Likelihood Estimates (MLEs) using the CODEML program. Use of these

parameters assumes that all the families evolve under the same model $model = 0$ and $NSsites = 2$, also known as the selection model for detecting those sites under positive selection. A more appropriate approach would be to fit all the NSsites models (1, 2, 3, 7 and 8) in a single run and compare the log-likelihood values for those models, however, given the large size of the dataset this would have resulted in prohibitive computational times. The likelihood of the models fitting the sequence data can be evaluated using the data provided on the PANDIT-PLUS (Dimitrieva and Anisimova, 2010) database in which the log-likelihoods of all the PFAM-A families have been provided. The challenge of utilising the PANDIT-PLUS database is that this database was last updated in 2005 and the PFAM-A database has undergone some extensive changes since then.

5.10 The inference of the domain function of the HBx protein

The HBx ω MLE obtained from the CODEML *rst* file were aligned against all the PFAM family ω profiles on the EvoDB database as described in Section 3.9. The top five statistically significant results are presented in Section 4.8.2 in Table 4.8. The provided results include only those families that were statistically significant and due to the scope of the research analysis was concluded at identifying the proteins and their functions. The objective of sequence alignment is to reduce the search space, such that resources can be invested in following up those inferences made by a sequence alignment algorithm. The results presented here can be confirmed by experimental methods and those are beyond the scope of this study.

The new algorithm, BLOSUM-FIRE, is provided to infer the domain functions of proteins, to achieve this, query sequences are aligned against the evolutionary profiles of PFAM-A protein families. The HBx was chosen because the inference of its role in viral replication has been elusive as reviewed in Section 2.15. The results (Table

4.8) show that the new algorithm was able to identify the domain family to which the HBx protein belongs to, the trans-activation X family. These results demonstrate the efficacy of the BLOSUM-FIRE algorithm at inferring statistically significant ($p = 0.0$) domain functions of the HBx protein. Analysis of these results reveals that based on the similarity of evolutionary rates profiles, this match would not have been found. Therefore, these results indicate the robustness of the new scoring scheme based on the evolutionary rate and BLOSUM scoring matrix approaches. The next statistically significant ($p = 7.6 \times 10^{-5}$) match found by FIRE and assessed using BLAST (Figure 4.26) was the rubella endopeptidase family with a relatively poor E-value score of 0.52. Furthermore, the BLAST found the alignment between HBx and rubella endopeptidase to be statistically significant with an E-value of 0.52 and a bit score of 15. Such an E-value is relatively high compared to the widely accepted E-values of 1×10^{-6} or below used for homology inference.

Analysis of the recorded functions of these proteins revealed similarities, for example, both proteins have been identified to play roles in viral replication, in the rubella virus the endopeptidase is required for processing of the virus replication protein. The Rubella virus (RB), an RNA virus is a member of the Togaviridae genus family and infections by this virus leads to the German measles a childhood disease endemic to all parts of the world (Zheng et al., 2003). The endopeptidase is the only non-structural protease encoded by the virus.

This non-structural protease classified under the family of papain-like proteases has been identified to play a role in trans- and cis- cleavage. The protease also has a conserved domain which is also called the X domain which has been identified by sequence comparison approaches. Regions of this X domain have been demonstrated in vitro to be required for cis-cleavage processes (Liang et al., 2000). This non-structural protease has been the focus of some studies (ten Dam et al., 1999). Analysis was also carried

using the web server also identified shared motifs between the proteins. The analysis revealed that the two proteins shared three similar motifs, although motif similarity is not an accepted approach for inferring functions it highlight some structural similarities shared by proteins that a residue based approach such as BLOSUM-IRE would not be able to detect. The results of the MEME analysis revealed that the proteins shared three statistically significant motif distributed along the two sequences, although such information cannot be used to conclude shared functions it demonstrated that the two proteins have regions of similarity.

5.11 The efficacy of using residues matched as a measure of performance

All of the experiments described here are evaluated using sequence identity. Whilst the novel approach of the algorithm is to detect distantly related proteins using evolutionary rates, it may not be appropriate to use the number of matched residues as a measure of performance it must not be overlooked that two unrelated protein may be under similar evolutionary rates due to chance alone and such occurrences result in false positive the residue score may be used a reality check for sequence alignment. The amino acids also determine the functional role or structural folds that may be in the amino acids. The use of structural based methods to validate the performance of the new algorithm would have been more appropriate and has been advocated for by Pei (2008), however, the input files required by the BLOSUM-FIRE algorithm results in alignments that do not have structures in the PDB database (Bernstein et al., 1977). Additionally, gold standard reference alignment benchmarks such as BaliBASE (Thompson et al., 2005), HOMSTRAD (Mizuguchi et al., 1998) and SABmark (Van Walle et al., 2005) would have been ideal, however, the algorithm requires orthologous sequences to calculate the ω MLEs at codon sites. One of the challenges faced with using the algorithm was the requirement of orthologous sequence to calculate ω profiles and additionally the

requirement of the nucleotide sequences.

Moreover, the HBx structure has not been determined, therefore a structural approach would not be applicable. In such cases, algorithms such as BLOSUM-FIRE can provide more insight than residue based approaches such as CLUSTAL Omega (Sievers et al., 2014) and MAFFT (Katoh and Toh, 2008).

5.12 The utility of evolutionary rate coupled approaches to sequence alignment

This study presents an algorithm based on an approach where an evolutionary based alignment approach has been coupled to a conventional amino acid based approach in a dynamic scoring function. This section discusses related studies where a coupled approach has been utilised in sequence alignment although not similar to the approach taken here. There are several tools which use the evolutionary rate to infer the functions of proteins of interest, however, it is not used as an alignment metric. These tools include the Evolutionary Trace method, Consurf and the Rate4Site tool. In the ET method (Lichtarge et al., 1996) using conservation information inferred from the evolutionary rate the information is to identify the 3D structure of proteins. Consurf on the other hand utilises the evolutionary rate to elucidate functions of protein (Celniker et al., 2013). Additionally, the Rate4Site algorithm utilises evolutionary information utilised the 3D structures coupled with a ML approach of measuring the evolutionary rate from alignments to identify functionally important regions, for example, binding grooves in the Src SH2 binding domains (Pupko et al., 2002). The above studies provide evidence that conservation inferred from the evolutionary rate can be effective at inferring function.

5.13 Applications of the BLOSUM-FIRE algorithm and EvoDB

The inference of query protein sequences is one of the most common tasks in molecular biology, however, difficulties arise if significant alignments cannot be made due to low sequence similarity. The study provides the BLOSUM-FIRE algorithm written in the Python programming language portable to any operating system. Finally, the BLOSUM-FIRE algorithm can be used to infer the protein domain functions of an unknown query sequences by searching the EvoDB. It is able to detect distantly related proteins using the evolutionary rate at codon sites. The algorithm produces alignments with qualities at par with conventional algorithms such as CLUSTAL Omega, MAFFT, MUSCLE and T-COFFEE.

5.14 Algorithm limitations

The BLOSUM-FIRE algorithm was conceptualised to address the challenge of low sequence similarity. It addresses the problems in the "twilight zone" of sequence alignment described by Rost (1999), where at regions of around 30% sequence similarity there is not enough evidence to infer homology. In this study, the BLOSUM-FIRE algorithm is provided to address this challenge, it uses a robust approach of an evolutionary rate based approach coupled with an conventional BLOSUM based approach to address the challenge of aligning sequences in the twilight zone. In the proceeding sections the limitations and challenges of using the algorithm and EvoDB database are discussed.

To generate an alignment; the evolutionary rates (ω) MLES at codon sides are required. To accurately determine these evolutionary rates orthologous MSA of the nucleotide sequences of the query sequence are required as described in section 3.2. It is important to note that while the algorithm generates amino acid alignments the

nucleotide sequences of the proteins must be available as they are required for the determination of the ω MLEs at codon sites using the CODEML program. The requirement of orthologous sequences means that sequences from different species would be required (Kuzniar et al., 2008). In some cases if a researchers is struggling to infer the homology of a novel protein due to low sequence similarity then finding corresponding sequences in other species may also be a challenge. Such a challenge may reduce the utility of the algorithm therefore, the algorithm is provided for specialized use cases.

The algorithm presented here uses the BEB MLE of ω calculated by the CODEML program and its models for sequence evolution. However, the calculation of the evolutionary rate profiles presents several challenges as identified by Yang (1996) and Yang et al. (2005). Therefore. it is possible of some of these inaccurate measures may result in inaccurate evolutionary profiles and subsequently wrong inferences may be made.

The BLOSUM-FIRE scoring function uses selective pressure to adjust BLOSUM scores, in which a an assumption is made that highly conserved sites are of functional importance and those site that are poorly conserved are neutrally evolving or under positive selection. This assumption is based on a generally accepted theory that those functionally important sites can be identified by measuring the evolutionary rate and this is the basis of several tools (Stern et al., 2007; Celniker et al., 2013; Pupko et al., 2002). This assumption may be flawed as some studies have provided evidence of functionally important sites which are poorly conserved in related sequence (Cooper and Brown, 2008). In such cases alignment algorithms using additional information such as secondary structure would prove to be more efficient in detecting shared functions as reviewed by Nei et al. (1983).

The determination of the ω MLEs at codon sites from sequence data and phylogenetic trees is a computationally intensive process; time and space requirements scale

with the length and numbers of sequences analysed. Therefore, it is possible that for sequences of certain numbers or length, the computational requirements may be prohibitive. Such a challenge resulted in the implementation of the EvoDB using a cluster based system as described in Section 3.8, however, such infrastructure may not be available to everyone. Such challenges can better be addressed by using an online server similar to the approach used for the PFAM database (Bateman, 2004) or BLAST (Altschul et al., 1990).

The implementation of the algorithm utilises a modified Needleman-Wunsch dynamic programming approach which finds global regions of similarity. Therefore, for sequences with local regions of similarity the FIRE approach would not be an appropriate choice. The challenge is that such a situation is inevitable without *a priori* knowledge of sequences. The global approach also means that sequence length divergence will affect the score of the final alignment and the greater this divergence the less sensitive the FIRE approach is.

5.15 Study Limitations

This section discusses the limitations of this study, some of these are related to the algorithm limitations already discussed in the previous section. Firsty, the study used a relatively small sample size to validate the approach. Consequently they may be scenarios where the hypothesis may be false. Secondly, the study does not an provide experiments on the false positive rate and whether the modifications made on the algorithm have resulted in statistical significant improvements. The widely accepted approache to address issues associated with problems of false positives is the use of receiver operating characteristic (ROC) (Gribskov and Robinson, 1996). However, the challenge is identifying good test cases, whilst benchmarks such as BAliBASE (Thompson et al., 2005) exist the challenge is that the FIRE approach requires nucleotide sequences to determine ω MLEs before analysis can be carried out. This requirement

of nucleotide sequences requires the conceptualisation of novel approaches to evaluate the poor sensitivity and specificity of the algorithm. Thirdly, the study did not utilise structural data (Bernstein et al., 1977; Murzin et al., 1995; Orengo et al., 1997) to obtain reference alignments, structural approaches are more effective than residue based approaches in detecting share functions. The challenge is that the algorithm uses profiles of families to infer protein domain function and each family consists of several protein members of different structures. Therefore, a framework to move from domain family level to protein member family is required, however, this requires additional analysis tools or approaches. Lastly, the study did not evaluate the accuracy of the CODEML program, the challenges of using the models to measure rates of evolution have been evaluated (Goldman and Yang, 1994). It is possible that the data obtained from CODEML may not be accurate, therefore resulting in poor alignments.

Some of the inherent limitations of the BLOSUM-FIRE algorithm and subsequently this study are the assumptions made when measuring positive selection under the M2a model using the CODEML program in the PAML suite. One of these assumptions is that for those sites under positive selection there have been numerous substitutions at that site across the phylogeny (Yang et al., 2005), however, the selection pressure may vary across the lineages for example in HIV (Guindon et al., 2004). Additionally, it is assumed that the non-synonymous substitution rate value varies at codon sites while the same value for synonymous substitution rate is used for all the codon sites, this assumption may be violated in real data and lead to the method failing (Pond and Muse, 2005).

5.16 Future directions

This section looks at specific areas that require refinement and improvement. One of the challenges that user would face using BLOSUM-FIRE algorithm is that the algorithm has a running time of at least 10 hours to search EvoDB depending on the length of

the query sequence. This is purely a performance issue and can be solved by analysing and profiling the memory usage and the execution of the algorithm and finding those parts of the program that can be sped up by implementing better approaches. By deploying the alignments tasks on a distributed system such as a cluster running times can be reduced to minutes. The final task would be to make the FIRE algorithm an on-line server where users can query the database similar to how the BLAST algorithm searches the GenBank database.

Chapter 6

Concluding remarks

An implementation of the BLOSUM-FIRE algorithm is presented, the algorithm utilises a scoring function that uses a BLOSUM62 based approach and an evolutionary rate based approach. The BLOSUM scaled approach uses the scores at aligned residues to score alignments which are scaled using the conservation inferred from the ω parameter. The BLOSUM-FIRE approach uses the similarity of the evolutionary rates at codon sites and this is scaled using a proportionality constant to make it comparable to BLOSUM scores. These two approaches are coupled together to produce a dynamic scoring algorithm depending on the conservation at the sequence level to balance the scoring of the aligned residues.

The BLOSUM-FIRE algorithm has a better performance than FIRE as a result of increased specificity of the quality of alignment produced. The use of the evolutionary rate approach coupled to a BLOSUM scoring matrix has proved to be more sensitive and specific than an evolutionary rate approach resulting in a robust algorithm. The BLOSUM-FIRE algorithm performs as well as conventional algorithms with little or no differences in performance compared to such tools as CLUSTAL Omega, T-COFFEE and MAFFT. The BLOSUM-FIRE Python software, user information and resources for using the software are freely available at www.bioinf.wits.ac.za/software/fire.

The study provides the EvoDB flat file database which is a compilation of all of the PFAM-A protein domain database evolutionary rate profiles, nucleotide alignments, phylogenetic trees and annotation data. The EvoDB flat file database, release notes and resources for using the database are freely available at www.bioinf.wits.ac.za/software/fire/evodb. The evolutionary rate profiles consist of the BEB of the MLEs of d_N/d_S or ω determined by the CODEML program found on the PAML suite of software. To demonstrate the utility of this algorithm, the study has provided a case study of using the new BLOSUM-FIRE algorithm to infer the protein domain functions of the HBx protein.

While other researchers may argue that study of MSA has been stagnant for several years, nevertheless, the requirement for innovative alignment tools and approaches increases every day as sequence databases grow. Those methods that use only the raw sequence data are out-performed by innovative tools that incorporate protein structure. It is also probable that those approaches that incorporate evolutionary information may provide better alternatives to amino acid residue base approaches. After all the challenge of sequence alignment is to detect homology. Homology is shared ancestry, evolutionary based approaches such as BLOSUM-FIRE use the evolutionary rate a similarity metric, it can be forecasted that as methods for measuring evolutionary rates at codon sites are improved so will the performance of evolutionary rate based methods such as BLOSUM-FIRE.

References

Abecasis, G., Altshuler, D., Auton, A., Brooks, L., Durbin, R., Gibbs, R. A., Hurles, M. E., McVean, G. A., Bentley, D., Chakravarti, A. et al. (2010), A map of human genome variation from population-scale sequencing., *Nature* **467**(7319), 1061–1073.

Afonnikov, D. A. and Kolchanov, N. A. (2004), CRASP: a program for analysis of coordinated substitutions in multiple alignments of protein sequences, *Nucleic Acids Research* **32**(suppl 2), W64–W68.

Agrawal, A. and Huang, X. (2009), Pairwise statistical significance of local sequence alignment using multiple parameter sets and empirical justification of parameter set change penalty, *BMC Bioinformatics* **10**(Suppl 3), S1.

Allison, L., Wallace, C. S. and Yee, C. N. (1992), Minimum message length encoding, evolutionary trees and multiple-alignment, *in* 'System Sciences, 1992. Proceedings of the Twenty-Fifth Hawaii International Conference on', **1**, 663–674.

Altschul, S. F. (1991), Amino acid substitution matrices from an information theoretic perspective, *Journal of Molecular Biology* **219**(3), 555–565.

Altschul, S. F. and Erickson, B. W. (1986), Optimal sequence alignment using affine gap costs, *Bulletin of Mathematical Biology* **48**(5), 603–616.

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.

Altschul, S., Gish, W., Miller, W., Myers, E. and Lipman, D. (1990), Basic Local Alignment Search Tool, *Journal of Molecular Biology* **215**, 403–410.

Angermüller, C., Biegert, A. and Söding, J. (2012), Discriminative modelling of context-specific amino acid substitution probabilities, *Bioinformatics* **28**(24), 3240–3247.

Anisimova, M., Bielawski, J. P. and Yang, Z. (2001), Accuracy and Power of the Likelihood Ratio Test in Detecting Adaptive Molecular Evolution, *Molecular Biology and Evolution* **18**(8), 1585–1592.

Antunes, A. and Ramos, M. J. (2007), Gathering computational genomics and proteomics to unravel adaptive evolution, *Evolutionary Bioinformatics Online* **3**, 207.

Bailey, T. L. and Elkan, C. (1994), Fitting a mixture model by expectation maximization to discover motifs in biopolymers, *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology* pp. 28–36.

Baldi, P., Chauvin, Y., Hunkapiller, T. and McClure, M. A. (1994), Hidden Markov models of biological primary sequence information., *Proceedings of the National Academy of Sciences* **91**(3), 1059–1063.

Barton, G. J. and Sternberg, M. J. (1987), Evaluation and improvements in the automatic alignment of protein sequences, *Protein Engineering* **1**(2), 89–94.

Bateman, A. (2004), The Pfam protein families database, *Nucleic Acids Research* **32**(90001), 138D–141D.

Bennet, S., Cohen, M. A. and Gonnet, G. H. (1994), Amino acid substitution during functionally constrained divergent evolution of protein sequences, *Protein Engineering* **7**(11), 1323–1332.

Benson, D. A., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J. and Sayers, E. W. (2010), GenBank, *Nucleic Acids Research* **38**(suppl 1), D46–D51.

Berger, M. and Munson, P. J. (1991), A novel randomized iterative strategy for aligning multiple protein sequences, *Computer Applications in the Biosciences: CABIOS* **7**(4), 479–484.

Bernstein, F. C., Koetzle, T. F., Williams, G. J., Meyer Jr, E. F., Brice, M. D., Rodgers, J. R., Kennard, O., Shimanouchi, T. and Tasumi, M. (1977), The Protein Data Bank: a computer-based archival file for macromolecular structures, *Journal of Molecular Biology* **112**(3), 535–542.

Bork, P. (1991), Shuffled domains in extracellular proteins, *FEBS letters* **286**(1), 47–54.

Bouchard, M. J. and Schneider, R. J. (2004), The enigmatic X gene of hepatitis B virus, *Journal of Virology* **78**(23), 12725–12734.

Bray, N. (2002), AVID: A Global Alignment Program, *Genome Research* **13**(1), 97–102.

Brenner, S. E., Chothia, C. and Hubbard, T. J. (1998), Assessing sequence comparison methods with reliable structurally identified distant evolutionary relationships, *Proceedings of the National Academy of Sciences* **95**(11), 6073–6078.

Brick, K. and Pizzi, E. (2008), A novel series of compositionally biased substitution matrices for comparing Plasmodium proteins, *BMC Bioinformatics* **9**(1), 236.

Brudno, M., Chapman, M., Göttgens, B., Batzoglou, S. and Morgenstern, B. (2003), Fast and sensitive multiple alignment of large genomic sequences, *BMC Bioinformatics* **4**(1), 66.

Brudno, M., Do, C. B., Cooper, G. M., Kim, M. F., Davydov, E., Green, E. D., Sidow, A., Batzoglou, S. et al. (2003), LAGAN and Multi-LAGAN: efficient tools for large-scale multiple alignment of genomic DNA, *Genome Research* **13**(4), 721–731.

Carrillo, H. and Lipman, D. (1988), The multiple sequence alignment problem in biology, *SIAM Journal on Applied Mathematics* **48**(5), 1073–1082.

Carroll, H., Ridge, P., Clement, M. and Snell, Q. (2006), Effects of gap open and gap extension penalties, *in* 'Proceedings of the Third Biotechnology and Bioinformatics Symposium. Brigham Young University, Utah', pp. 19–23.

Cavalli-Sforza, L. L. and Edwards, A. W. (1967), Phylogenetic analysis. Models and estimation procedures, *American Journal of Human Genetics* **19**(3 Pt 1), 233.

Celniker, G., Nimrod, G., Ashkenazy, H., Glaser, F., Martz, E., Mayrose, I., Pupko, T. and Ben-Tal, N. (2013), ConSurf: Using Evolutionary Data to Raise Testable Hypotheses about Protein Function, *Israel Journal of Chemistry* **53**(3-4), 199–206.

Chang, S.-F., Netter, H. J., Hildt, E., Schuster, R., Schaefer, S., Hsu, Y.-C., Rang, A. and Will, H. (2001), Duck hepatitis B virus expresses a regulatory HBx-like protein from a hidden open reading frame, *Journal of Virology* **75**(1), 161–170.

Chothia, C. and Lesk, A. M. (1986), The relation between divergence of sequence and structure in proteins, *EMBO Journal* **5**, 823–826.

Clark, N. L. and Aquadro, C. F. (2009), A Novel Method to Detect Proteins Evolving at Correlated Rates: Identifying New Functional Relationships between Coevolving Proteins, *Molecular Biology and Evolution* **27**(5), 1152–1161.

Cock, P. J., Antao, T., Chang, J. T., Chapman, B. A., Cox, C. J., Dalke, A., Friedberg, I., Hamelryck, T., Kauff, F., Wilczynski, B. et al. (2009), Biopython: freely available Python tools for computational molecular biology and bioinformatics, *Bioinformatics* **25**(11), 1422–1423.

Colgrove, R., Simon, G. and Ganem, D. (1989), Transcriptional activation of homologous and heterologous genes by the hepatitis B virus X gene product in cells permissive for viral replication., *Journal of Virology* **63**(9), 4019–4026.

Consortium, U. et al. (2013), Activities at the Universal Protein Resource (UniProt), *Nucleic Acids Research* p. gkt1140.

Cooper, G. M. and Brown, C. D. (2008), Qualifying the relationship between sequence conservation and molecular function, *Genome Research* **18**(2), 201–205.

Coronado, J. E., Attie, O., Epstein, S. L., Qiu, W.-G. and Lipke, P. N. (2006), Composition-modified matrices improve identification of homologs of saccharomyces cerevisiae low-complexity glycoproteins, *Eukaryotic Cell* **5**(4), 628–637.

Corpet, F. (1988), Multiple sequence alignment with hierarchical clustering, *Nucleic Acids Research* **16**(22), 10881–10890.

Dayhoff, M. O. and Schwartz, R. M. (1978), Chapter 22: A model of evolutionary change in proteins, *in* 'in Atlas of Protein Sequence and Structure'.

Delport, W., Poon, A. F., Frost, S. D. and Pond, S. L. K. (2010), Datamonkey 2010: a suite of phylogenetic analysis tools for evolutionary biology, *Bioinformatics* **26**(19), 2455–2457.

Delport, W., Scheffler, K. and Seoighe, C. (2009), Models of coding sequence evolution, *Briefings in Bioinformatics* **10**(1), 97–109.

Di Tommaso, P., Moretti, S., Xenarios, I., Orobitg, M., Montanyola, A., Chang, J.-M., Taly, J.-F. and Notredame, C. (2011), T-Coffee: a web server for the multiple sequence alignment of protein and RNA sequences using structural information and homology extension, *Nucleic Acids Research* **39**(suppl 2), W13–W17.

Dimitrieva, S. and Anisimova, M. (2010), PANDITplus: toward better integration of evolutionary view on molecular sequences with supplementary bioinformatics resources, *Trends in Evolutionary Biology* **2**(1), e1.

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. (1981), Similar amino acid sequences: chance or common ancestry?, *Science* **214**(4517), 149–159.

Drummond, A. J. and Rambaut, A. (2007), BEAST: Bayesian evolutionary analysis by sampling trees, *BMC Evolutionary Biology* **7**(1), 214.

Durand, P., Hazelhurst, S. and Coetzer, T. (2010), Evolutionary rates at codon sites may be used to align sequences and infer protein domain function, *BMC bioinformatics* **11**(1), 151.

Edgar, R. C. (2004), MUSCLE: a multiple sequence alignment method with reduced time and space complexity, *BMC Bioinformatics* **5**(1), 113.

Edgar, R. C. (2009), Optimizing substitution matrix choice and gap parameters for sequence alignment, *BMC Bioinformatics* **10**(1), 396.

Edwards, R. J. and Shields, D. C. (2005), BADASP: predicting functional specificity in protein families using ancestral sequences, *Bioinformatics* **21**(22), 4190–4191.

Falquet, L., Pagni, M., Bucher, P., Hulo, N., Sigrist, C. J., Hofmann, K. and Bairoch, A. (2002), The PROSITE database, its status in 2002, *Nucleic Acids Research* **30**(1), 235–238.

Felsenstein, J. (1973), Maximum-likelihood estimation of evolutionary trees from continuous characters., *American Journal of Human Genetics* **25**(5), 471.

Felsenstein, J. (1978), Cases in which parsimony or compatibility methods will be positively misleading, *Systematic Biology* **27**(4), 401–410.

Felsenstein, J. (1981), Evolutionary trees from gene frequencies and quantitative characters: finding maximum likelihood estimates, *Evolution* pp. 1229–1242.

Felsenstein, J. (2006), PHYLIP (Phylogeny Inference Package) documentation files, version 3.66, *Seattle: Department of Genome Sciences, University of Washington* .

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, S., Jing, H., Zhong-xi, M. and Hui-rao, Z. (2002), A genetic algorithm on multiple sequences alignment problems in biology, *Wuhan University Journal of Natural Sciences* **7**(2), 139–144.

Finn, R. D., Clements, J. and Eddy, S. R. (2011), HMMER web server: interactive sequence similarity searching, *Nucleic Acids Research* **39**(suppl 2), W29–W37.

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., Margoliash, E. et al. (1967), Construction of phylogenetic trees, *Science* **155**(760), 279–284.

Ganem, D. and Varmus, H. (1987), The molecular biology of the hepatitis B viruses, *Annual Review of Biochemistry* **56**(1), 651–693.

Goldman, N. and Yang, Z. (1994), A codon-based model of nucleotide substitution for protein-coding DNA sequences., *Molecular Biology and Evolution* **11**(5), 725–736.

Gonnet, G. H., Korostensky, C. and Benner, S. (2000), Evaluation measures of multiple sequence alignments, *Journal of Computational Biology* **7**(1-2), 261–276.

Gonnet, G. and Korostensky, C. (1999), *Optimal scoring matrices for estimating distances between aligned sequences*, manuscript.

Gotoh, O. (1982), An improved algorithm for matching biological sequences, *Journal of Molecular Biology* **162**(3), 705–708.

Gotoh, O. (1990), Optimal sequence lignment allowing for long gaps, *Bulletin of Mathematical Biology* **52**(3), 359–373.

Gotoh, O. (1995), A weighting system and aigorithm for aligning many phylogenetically related sequences, *Computer applications in the biosciences: CABIOS* **11**(5), 543–551.

Gotoh, O. (1996), Significant improvement in accuracy of multiple protein sequence alignments by iterative refinement as assessed by reference to structural alignments, *Journal of Molecular Biology* **264**(4), 823–838.

Gribskov, M., McLachlan, A. D. and Eisenberg, D. (1987), Profile analysis: detection of distantly related proteins, *Proceedings of the National Academy of Sciences* **84**(13), 4355–4358.

Gribskov, M. and Robinson, N. L. (1996), Use of receiver operating characteristic (ROC) analysis to evaluate sequence matching, *Computers & chemistry* **20**(1), 25–33.

Guindon, S., Dufayard, J.-F., Lefort, V., Anisimova, M., Hordijk, W. and Gascuel, O. (2010), New algorithms and methods to estimate maximum-likelihood phylogenies: assessing the performance of PhyML 3.0, *Systematic biology* **59**(3), 307–321.

Guindon, S., Rodrigo, A. G., Dyer, K. A. and Huelsenbeck, J. P. (2004), Modeling the site-specific variation of selection patterns along lineages, *Proceedings of the National Academy of Sciences of the United States of America* **101**(35), 12957–12962.

Hamosh, A., Scott, A. F., Amberger, J. S., Bocchini, C. A. and McKusick, V. A. (2005), Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders, *Nucleic Acids Research* **33**(suppl 1), D514–D517.

Hasegawa, M., Kishino, H. and Yano, T.-a. (1985), Dating of the human-ape splitting by a molecular clock of mitochondrial DNA, *Journal of Molecular Evolution* **22**(2), 160–174.

Henikoff, S. and Henikoff, J. G. (1992), Amino acid substitution matrices from protein blocks, *Proceedings of the National Academy of Sciences* **89**(22), 109–115.

Henikoff, S. and Henikoff, J. G. (1993), Performance evaluation of amino acid substitution matrices, *Proteins: Structure, Function, and Bioinformatics* **17**(1), 49–61.

Higgins, D. G. and Sharp, P. M. (1988), CLUSTAL: a package for performing multiple sequence alignment on a microcomputer, *Gene* **73**(1), 237–244.

Hirosawa, M., Totoki, Y., Hoshida, M. and Ishikawa, M. (1995), Comprehensive study on iterative algorithms of multiple sequence alignment, *Computer Applications in the Biosciences: CABIOS* **11**(1), 13–18.

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., Kääriäinen, S., Rosenström, P. and Schenkel, A. (2008), Searching protein structure databases with DaliLite v. 3, *Bioinformatics* **24**(23), 2780–2781.

Holm, L. and Sander, C. (1994), The FSSP database of structurally aligned protein fold families., *Nucleic Acids Research* **22**(17), 3600.

Holmes, I. and Bruno, W. J. (2001), Evolutionary HMMs: a Bayesian approach to multiple alignment, *Bioinformatics* **17**(9), 803–820.

Huang, J. Y. and Brutlag, D. L. (2001), The EMOTIF database, *Nucleic Acids Research* **29**(1), 202–204.

Huelsenbeck, J. P. and Crandall, K. A. (1997), Phylogeny estimation and hypothesis testing using maximum likelihood, *Annual Review of Ecology and Systematics* pp. 437–466.

Huelsenbeck, J. P., Jain, S., Frost, S. W. and Pond, S. L. (2006), A Dirichlet process model for detecting positive selection in protein-coding DNA sequences, *Proceedings of the National Academy of Sciences* **103**(16), 6263–6268.

Jones, D. T. (1999), Protein secondary structure prediction based on position-specific scoring matrices, *Journal of Molecular Biology* **292**(2), 195–202.

Jones, D. T., Taylor, W. R. and Thornton, J. M. (1992), The rapid generation of mutation data matrices from protein sequences, *Computer Applications in the Biosciences: CABIOS* **8**(3), 275–282.

Jørgensen, F. G., Hobolth, A., Hornshøj, H., Bendixen, C., Fredholm, M. and Schierup, M. H. (2005), Comparative analysis of protein coding sequences from human, mouse and the domesticated pig, *BMC Biology* **3**(1), 2.

Jukes, T., Cantor, C. and Munro, H. (1969), Mammalian protein metabolism, *Evolution of Protein Molecules* **3**, 121–132.

Kaminuma, E., Kosuge, T., Kodama, Y., Aono, H., Mashima, J., Gojobori, T., Sugawara, H., Ogasawara, O., Takagi, T., Okubo, K. et al. (2011), DDBJ progress report, *Nucleic Acids Research* **39**(suppl 1), D22–D27.

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.

Karplus, K., Barrett, C. and Hughey, R. (1998), Hidden Markov models for detecting remote protein homologies., *Bioinformatics* **14**(10), 846–856.

Katoh, K. and Toh, H. (2008), Recent developments in the MAFFT multiple sequence alignment program, *Briefings in Bioinformatics* **9**(4), 286–298.

Kimura, M. (1980), A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences, *Journal of molecular evolution* **16**(2), 111–120.

Kimura, M. (1984), *The neutral theory of molecular evolution*, Cambridge University Press.

Krogh, A., Brown, M., Mian, I. S., Sjlander, K. and Haussler, D. (1994), Hidden Markov models in computational biology: Applications to protein modeling, *Journal of Molecular biology* **235**(5), 1501–1531.

Kumar, M., Jung, S. Y., Hodgson, A. J., Madden, C. R., Qin, J. and Slagle, B. L. (2011), Hepatitis B virus regulatory HBx protein binds to adaptor protein IPS-1 and inhibits the activation of beta interferon, *Journal of virology* **85**(2), 987–995.

Kumar, S., Tamura, K. and Nei, M. (2004), MEGA3: integrated software for molecular evolutionary genetics analysis and sequence alignment, *Briefings in Bioinformatics* **5**(2), 150–163.

Kuzniar, A., Van Ham, R. C., S., P. and Leunissen, J. A. (2008), The quest for orthologs:finding the corresponding gene across genomes., *Trends in Genetics* **24(11)**, 539–551.

Lander, E. S., Linton, L. M., Birren, B., Nusbaum, C., Zody, M. C., Baldwin, J., Devon, K., Dewar, K., Doyle, M., FitzHugh, W. et al. (2001), Initial sequencing and analysis of the human genome, *Nature* **409**(6822), 860–921.

Lemaitre, C., Barré, A., Citti, C., Tardy, F., Thiaucourt, F., Sirand-Pugnet, P. and Thébault, P. (2011), A novel substitution matrix fitted to the compositional bias in Mollicutes improves the prediction of homologous relationships, *BMC Bioinformatics* **12**(1), 457.

Liang, H., Zhou, W. and Landweber, L. F. (2006), SWAKK: a web server for detecting positive selection in proteins using a sliding window substitution rate analysis, *Nucleic acids research* **34**(suppl 2), W382–W384.

Liang, T. J. (2009), Hepatitis B: the virus and disease, *Hepatology* **49**(S5), S13–S21.

Liang, Y., Yao, J. and Gillam, S. (2000), Rubella virus nonstructural protein protease domains involved in trans-and cis-cleavage activities, *Journal of virology* **74**(12), 5412–5423.

Lichtarge, O., Bourne, H. R. and Cohen, F. E. (1996), An evolutionary trace method defines binding surfaces common to protein families, *Journal of Molecular Biology* **257**(2), 342–358.

Liu, Y., Schmidt, B. and Maskell, D. L. (2010), MSAProbs: multiple sequence alignment based on pair hidden Markov models and partition function posterior probabilities, *Bioinformatics* **26**(16), 1958–1964.

Lunter, G., Mikls, I., Drummond, A., Jensen, J. L. and Hein, J. (2005), Bayesian coestimation of phylogeny and sequence alignment, *BMC Bioinformatics* **6**(1), 83.

Madden, C. R. and Slagle, B. L. (2001), Stimulation of cellular proliferation by hepatitis B virus X protein, *Disease Markers* **17**(3), 153–157.

Mistry, J., Finn, R. D., Eddy, S. R., Bateman, A. and Punta, M. (2013), Challenges in homology search: HMMER3 and convergent evolution of coiled-coil regions, *Nucleic Acids Research* .

Miyata, T. and Yasunaga, T. (1980), Molecular evolution of mRNA: a method for estimating evolutionary rates of synonymous and amino acid substitutions from homologous nucleotide sequences and its application, *Journal of Molecular Evolution* **16**(1), 23–36.

Mizuguchi, K., Deane, C. M., Blundell, T. L. and Overington, J. P. (1998), HOMSTRAD: a database of protein structure alignments for homologous families, *Protein Science* **7**(11), 2469–2471.

Morgenstern, B., Frech, K., Dress, A. and Werner, T. (1998), DIALIGN: finding local similarities by multiple sequence alignment., *Bioinformatics* **14**(3), 290–294.

Murakami, S. (1999), Hepatitis B virus X protein: structure, function and biology, *Intervirology* **42**(2-3), 81–99.

Murata, M., Richardson, J. S. and Sussman, J. L. (1985), Simultaneous comparison of three protein sequences, *Proceedings of the National Academy of Sciences* **82**(10), 3073–3077.

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.

Muse, S. V. and Gaut, B. S. (1994), A likelihood approach for comparing synonymous and nonsynonymous nucleotide substitution rates, with application to the chloroplast genome., *Molecular Biology and Evolution* **11**(5), 715–724.

Myers, E. W. and Miller, W. (1988), Optimal alignments in linear space, *Computer Applications in the Biosciences: CABIOS* **4**(1), 11–17.

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 Gojobori, T. (1986), Simple methods for estimating the numbers of synonymous and nonsynonymous nucleotide substitutions., *Molecular Biology and Evolution* **3**(5), 418–426.

Nei, M., Tajima, F. and Tateno, Y. (1983), Accuracy of estimated phylogenetic trees from molecular data, *Journal of Molecular Evolution* **19**(2), 153–170.

Nekrutenko, A., Makova, K. D. and Li, W.-H. (2002), The KA/KS ratio test for assessing the protein-coding potential of genomic regions: an empirical and simulation study, *Genome Research* **12**(1), 198–202.

Ngandu, N. K., Scheffler, K., Moore, P., Woodman, Z., Martin, D., Seoighe, C. et al. (2008), Extensive purifying selection acting on synonymous sites in HIV-1 Group M sequences, *Virology Journal* **5**, 160.

Nielsen, R. and Yang, Z. (1998), Likelihood models for detecting positively selected amino acid sites and applications to the HIV-1 envelope gene, *Genetics* **148**(3), 929–936.

Notredame, C. (2002), Recent progress in multiple sequence alignment: a survey, *Pharmacogenomics* **3**(1), 131–144.

Notredame, C., Holm, L. and Higgins, D. G. (1998), COFFEE: an objective function for multiple sequence alignments., *Bioinformatics* **14**(5), 407–422.

Nuin, P. A., Wang, Z. and Tillier, E. R. (2006), The accuracy of several multiple sequence alignment programs for proteins, *BMC Bioinformatics* **7**(1), 471.

Orengo, C. A., Jones, D. T. and Thornton, J. M. (1994), Protein superfamilies and domain superfolds, *Nature* **372**(6507), 631–634.

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.

Orengo, C. A., Todd, A. E. and Thornton, J. M. (1999), From protein structure to function, *Current Opinion in Structural Biology* **9**(3), 374–382.

O'Sullivan, O., Zehnder, M., Higgins, D., Bucher, P., Grosdidier, A. and Notredame, C. (2003), APDB: a novel measure for benchmarking sequence alignment methods without reference alignments, *Bioinformatics* **19**(suppl 1), i215–i221.

Overington, J., Donnelly, D., Johnson, M. S., Sali, A. and Blundell, T. L. (1992), Environment-specific amino acid substitution tables: Tertiary templates and prediction of protein folds, *Protein Science* **1**(2), 216–226.

Paila, U., Kondam, R. and Ranjan, A. (2008), Genome bias influences amino acid choices: analysis of amino acid substitution and re-compilation of substitution matrices exclusive to an AT-biased genome, *Nucleic Acids Research* **36**(21), 6664–6675.

Pearson, W. R. (1995), Comparison of methods for searching protein sequence databases, *Protein Science* **4**(6), 1145–1160.

Pearson, W. R. (1998), Empirical statistical estimates for sequence similarity searches, *Journal of Molecular Biology* **276**(1), 71–84.

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.

Pei, J. (2008), Multiple protein sequence alignment, *Current Opinion in Structural Biology* **18**(3), 382–386.

Pond, S. L. K., Scheffler, K., Gravenor, M. B., Poon, A. F. and Frost, S. D. (2010), Evolutionary fingerprinting of genes, *Molecular biology and evolution* **27**(3), 520–536.

Pond, S. L. and Muse, S. V. (2005), HyPhy: hypothesis testing using phylogenies, *Statistical Methods in Molecular Evolution* pp. 125–181.

Posada, D. and Buckley, T. R. (2004), Model Selection and Model Averaging in Phylogenetics: Advantages of Akaike Information Criterion and Bayesian Approaches Over Likelihood Ratio Tests, *Systematic Biology* **53**(5), 793–808.

Price, M. N., Dehal, P. S. and Arkin, A. P. (2010), FastTree 2–approximately maximum-likelihood trees for large alignments, *PloS One* **5**(3), e9490.

Pupko, T., Bell, R. E., Mayrose, I., Glaser, F. and Ben-Tal, N. (2002), Rate4Site: an algorithmic tool for the identification of functional regions in proteins by surface mapping of evolutionary determinants within their homologues, *Bioinformatics* **18**(suppl 1), S71–S77.

Raghava, G. P. S. and Barton, G. J. (2006), Quantification of the variation in percentage identity for protein sequence alignments, *BMC Bioinformatics* **7**(1), 415.

Raines, R. (1998), Gene translocation links insects and crustaceans, *Nature* **392**, 667.

Reese, J. and Pearson, W. R. (2002), Empirical determination of effective gap penalties for sequence comparison, *Bioinformatics* **18**(11), 1500–1507.

Rice, P., Longden, I. and Bleasby, A. (2000), EMBOSS: the European molecular biology open software suite, *Trends in Genetics* **16**(6), 276–277.

Ronquist, F., Teslenko, M., van der Mark, P., Ayres, D. L., Darling, A., Höhna, S., Larget, B., Liu, L., Suchard, M. A. and Huelsenbeck, J. P. (2012), MrBayes 3.2: efficient Bayesian phylogenetic inference and model choice across a large model space, *Systematic Biology* **61**(3), 539–542.

Roshan, U. and Livesay, D. R. (2006), Probalign: multiple sequence alignment using partition function posterior probabilities, *Bioinformatics* **22**(22), 2715–2721.

Rost, B. (1999), Twilight zone of protein sequence alignments, *Protein Engineering* **12**(2), 85–94.

Rzhetsky, A. and Nei, M. (1995), Tests of applicability of several substitution models for DNA sequence data., *Molecular Biology and Evolution* **12**(1), 131–151.

Sadri, J., Diallo, A. B. and Blanchette, M. (2011), Predicting site-specific human selective pressure using evolutionary signatures, *Bioinformatics* **27**(13), i266–i274.

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

Sankoff, D. (1972), Matching sequences under deletion/insertion constraints, *Proceedings of the National Academy of Sciences* **69**(1), 4–6.

Sankoff, D. (1975), Minimal mutation trees of sequences, *SIAM Journal on Applied Mathematics* **28**(1), 35–42.

Schwartz, S., Kent, W. J., Smit, A., Zhang, Z., Baertsch, R., Hardison, R. C., Haussler, D. and Miller, W. (2003), Human–mouse alignments with BLASTZ, *Genome Research* **13**(1), 103–107.

Sellers, P. H. (1974), An algorithm for the distance between two finite sequences, *Journal of Combinatorial Theory, Series A* **16**(2), 253–258.

Shoemaker, B. A., Panchenko, A. R. and Bryant, S. H. (2006), Finding biologically relevant protein domain interactions: conserved binding mode analysis, *Protein Science* **15**(2), 352–361.

Sievers, F., Wilm, A., Dineen, D., Gibson, T. J., Karplus, K., Li, W., Lopez, R., McWilliam, H., Remmert, M., Soding, J., Thompson, J. D. and Higgins, D. G. (2014), Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega, *Molecular Systems Biology* **7**(1), 539–539.

Smith, T. F. and Waterman, M. S. (1981), Identification of common molecular subsequences, *Journal of Molecular Biology* **147**(1), 195–197.

Sneath, P. H., Sokal, R. R. et al. (1973), *Numerical taxonomy. The principles and practice of numerical classification.*

Söding, J. (2005), Protein homology detection by HMM–HMM comparison, *Bioinformatics* **21**(7), 951–960.

Sokal, R. R. and Michener, C. D. (1958), *A statistical method for evaluating systematic relationships*, University of Kansas.

Stamatakis, A. (2006), RAxML-VI-HPC: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models, *Bioinformatics* **22**(21), 2688–2690.

Stern, A., Doron-Faigenboim, A., Erez, E., Martz, E., Bacharach, E. and Pupko, T. (2007), Selecton 2007: advanced models for detecting positive and purifying selection using a Bayesian inference approach, *Nucleic Acids Research* **35**(Web Server), W506–W511.

Stoesser, G., Baker, W., van den Broek, A., Camon, E., Garcia-Pastor, M., Kanz, C., Kulikova, T., Leinonen, R., Lin, Q., Lombard, V. et al. (2002), The EMBL nucleotide sequence database, *Nucleic Acids Research* **30**(1), 21–26.

Studier, J. A. and Keppler, K. J. (1988), A note on the neighbor-joining algorithm of Saitou and Nei, *Molecular Biology and Evolution* **5**(6), 729–731.

Swofford, D. L. (2003), *PAUP*. Phylogenetic Analysis Using Parsimony (*and Other Methods). Version 4.*, Sinauer Associates, Sunderland, Massachusetts.

Tajima, F. and Nei, M. (1984), Estimation of evolutionary distance between nucleotide sequences., *Molecular Biology and Evolution* **1**(3), 269–285.

Tamura, K. (1992), Estimation of the number of nucleotide substitutions when there are strong transition-transversion and G+ C-content biases., *Molecular Biology and Evolution* **9**(4), 678–687.

Tamura, K. and Nei, M. (1993), Estimation of the number of nucleotide substitutions in the control region of mitochondrial DNA in humans and chimpanzees., *Molecular Biology and Evolution* **10**(3), 512–526.

Tavaré, S. (1986), Some probabilistic and statistical problems in the analysis of DNA sequences, *Lect. Math. Life Sci* **17**, 57–86.

Taylor, W. R. (1988), A flexible method to align large numbers of biological sequences, *Journal of Molecular Evolution* **28**(1-2), 161–169.

Taylor, W. R. and Orengo, C. A. (1989), Protein structure alignment, *Journal of Molecular Biology* **208**(1), 1 – 22.

ten Dam, E., Flint, M. and Ryan, M. D. (1999), Virus-encoded proteinases of the Togaviridae, *Journal of general virology* **80**(8), 1879–1888.

Terradillos, O., Pollicino, T., Lecoeur, H., Tripodi, M., Gougeon, M.-L., Tiollais, P. and Buendia, M. A. (1998), p53-independent apoptotic effects of the hepatitis B virus HBx protein in vivo and in vitro., *Oncogene* **17**(16), 2115–2123.

Thompson, J. D., Higgins, D. G. and Gibson, T. J. (1994), CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice, *Nucleic Acids Research* **22**(22), 4673–4680.

Thompson, J. D., Koehl, P., Ripp, R. and Poch, O. (2005), BAliBASE 3.0: latest developments of the multiple sequence alignment benchmark, *Proteins: Structure, Function, and Bioinformatics* **61**(1), 127–136.

Thompson, J. D., Plewniak, F. and Poch, O. (1999), A comprehensive comparison of multiple sequence alignment programs, *Nucleic Acids Research* **27**(13), 2682–2690.

Thompson, L. J. (2012), Recombinant Expression And Bioinformatic Analysis Of the Hepatitis B Virus X protein, PhD thesis, University of the Witwatersrand, Johannesburg.

Thorne, J. L., Kishino, H. and Felsenstein, J. (1991), An evolutionary model for maximum likelihood alignment of DNA sequences, *Journal of Molecular Evolution* **33**(2), 114–124.

Van Rossum, G. and Drake Jr, F. L. (1995), *Python reference manual*, Centrum voor Wiskunde en Informatica Amsterdam.

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.

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.

Wang, L. and Jiang, T. (1994), On the complexity of multiple sequence alignment, *Journal of Computational Biology* **1**(4), 337–348.

Waterman, M. S. and Perlwitz, M. D. (1984), Line geometries for sequence comparisons, *Bulletin of Mathematical Biology* **46**(4), 567–577.

Waterman, M. S. and Vingron, M. (1994), Rapid and accurate estimates of statistical significance for sequence data base searches, *Proceedings of the National Academy of Sciences* **91**(11), 4625–4628.

Whelan, S., de Bakker, P. I., Quevillon, E., Rodriguez, N. and Goldman, N. (2006), PANDIT: an evolution-centric database of protein and associated nucleotide domains with inferred trees, *Nucleic Acids Research* **34**(suppl 1), D327–D331.

Wu, C. H., Yeh, L.-S. L., Huang, H., Arminski, L., Castro-Alvear, J., Chen, Y., Hu, Z., Kourtesis, P., Ledley, R. S., Suzek, B. E. et al. (2003), The protein information resource, *Nucleic Acids Research* **31**(1), 345–347.

Xu, Z., Yen, T. B., Wu, L., Madden, C. R., Tan, W., Slagle, B. L. and Ou, J.-h. (2002), Enhancement of hepatitis B virus replication by its X protein in transgenic mice, *Journal of Virology* **76**(5), 2579–2584.

Yamada, K. and Tomii, K. (2013), Revisiting amino acid substitution matrices for identifying distantly related proteins, *Bioinformatics* .

Yang, Z. (1996), Phylogenetic analysis using parsimony and likelihood methods, *Journal of Molecular Evolution* **42**(2), 294–307.

Yang, Z. (1998), Likelihood ratio tests for detecting positive selection and application to primate lysozyme evolution., *Molecular Biology and Evolution* **15**(5), 568–573.

Yang, Z. (2007), PAML 4: phylogenetic analysis by maximum likelihood, *Molecular Biology and Evolution* **24**(8), 1586–1591.

Yang, Z. and Nielsen, R. (2000), Estimating synonymous and nonsynonymous substitution rates under realistic evolutionary models, *Molecular biology and evolution* **17**(1), 32–43.

Yang, Z. and Rannala, B. (2012), Molecular phylogenetics: principles and practice, *Nature Reviews Genetics* **13**(5), 303–314.

Yang, Z., Wong, W. S. and Nielsen, R. (2005), Bayes empirical Bayes inference of amino acid sites under positive selection, *Molecular Biology and Evolution* **22**(4), 1107–1118.

Ye, Y. and Godzik, A. (2003), Flexible structure alignment by chaining aligned fragment pairs allowing twists, *Bioinformatics* **19**(suppl 2), ii246–ii255.

Yen, T. B. (1996), Hepadnaviral X protein: review of recent progress, *Journal of Biomedical Science* **3**(1), 20–30.

Zheng, D.-P., Frey, T. K., Icenogle, J., Katow, S., Abernathy, E. S., Song, K.-J., Xu, W.-B., Yarulin, V., Desjatskova, R., Aboudy, Y. et al. (2003), Global distribution of rubella virus genotypes, *Emerging infectious diseases* **9**(12), 1523.

Zuckerkandl, E. and Pauling, L. (1965), Evolutionary divergence and convergence in proteins, *Evolving Genes and Proteins* **97**, 97–166.

Appendices

Appendix A: The FIRE algorithm Python file

This file is the Python source code for the BLOSUM-FIRE alignment program, *fire.py*, the algorithm requires a file of BLOSUM substitution matrices, *Blosum_matrix.py*, to run.

```
#!/bin/python
##############################################################################
#                          fire.py                                           #
#                                                                            #
#                    The BLOSUM-FIRE algorithm                               #
#                                                                            #
#                                                                            #
##############################################################################
from Blosum_matrix import do_blosum
from optparse import OptionParser
import random
import copy
import math
import sys
import re
import os

def InitStuff():

    usage = "usage: %prog [options] file1 file2"
    parser = OptionParser(usage=usage)
    parser.add_option("-r","--regexp",dest="regexp",
                  default = "m3",
                  metavar = "regexp",
                  help = "position of omega value"),
    parser.add_option("-w","--window_len",dest="window_len",
                  default = "4,8,16",
                  metavar = "INT,INT,INT",
                  help = "windows to be shown"),
    parser.add_option("-G","--gnuplot",dest="gnuplot",
                  default= False,
                  action = "store_true",
                  help = "Produce a codon versus similarity plot picture"),
    parser.add_option("-p","--plot",dest="plot_fname",
                  metavar = "FILE",
                  help = "plot as postscript  or user defined using -t/--gnu_term option"),
    parser.add_option("-t","--gnu_terminal",
                  dest="gnu_term",
                  default ='eps',
                  metavar = "terminal",
                  help = "set the gnuplot terminal")
    parser.add_option("-H","--histogram",dest="histo",
                  action = "store_true",
                  default= False,
                  help = "produce histogram"),
    parser.add_option("-b","--Blosum",dest="blosum",
                  default= 62 ,metavar="INT",
                  help = "BLOSUM SEQUENCE IDENTITY"),
    parser.add_option("-R","--random",dest="random",
                  default = "",
                  metavar = "INT",
```

```
                  help = "random trials")
    parser.add_option("-s","--seqalign",dest="seqalign",
                  action = "store_true",
                  default = False,
                  help = "produce alignment")
    parser.add_option("-f","--seqformat",dest="seqformat",
                  default = "fasta",
                  metavar = "SeqFormat",
                  help = "format for sequence files")
    parser.add_option("-o","--align_out",dest="seqout",
                  metavar = "alignment file",
                  help = "output for sequence files"),
    parser.add_option("-g","--gop",dest="gop",
                  default = False,
                  metavar = "FLOAT",
                  help = "gap open penalty")
    parser.add_option("-x","--gep",dest="gep",
                  default = False,
                  metavar = "FLOAT",
                  help = "gap extension penalty")
    parser.add_option("-q","--quiet",dest="quiet",
                  action = "store_true",
                  default = False,
                  help = "nothing on the screen")
    return parser


def get_filenames(args, parser):
    if len(args) != 2:
        parser.print_help()
        sys.exit(1)
    fname1, fname2 = args
    return fname1, fname2


def get_data(fname,regexp):

    """
        Get the filename with PAML output and return
        the corresponding proteins and the evolutionary
        rates specified by in the options
    """
    data = []
    f = open(fname)

    prot = ""
    for row in f:
        if "!!DATA" in row: break
        curr = []
        mm = re.search(regexp, row)
        if mm:
            aa  = mm.group(1)
            val = mm.group(2)
            curr.append(float(val))
        else:
            print "The row <%s> in the %s file is not in the right format "%(row,fname)
            sys.exit(1)
        data.append(curr)
        prot = prot+aa
    return (prot,data)


def Run_Blosum(matrix):

    """
       Get the Blosum matrix value from the imported Blosum_matrix.py module and get the
       blosums matrix, Blosum Variable (matrix) and the gap opening penalty(gop)
       and the gap extention penalty(gep)
    """
    # Biopython has dictionary for this
```

```
    # TO DO USE DICTIONARY METHOD IN BIOPYTHON

    return do_blosum(matrix)


def score_blosum(Blosum, aa1, aa2):

    """
        Use the Blosum matrix to get the blosum score for amino acid1(aa1) and the amino acid2(aa2)
    """
    Blosum = Blosum.split("\n")
    #Align the row and columns correctly fron the list to string

    for row in Blosum:
        if re.match(aa1,row[0]):
            try:
                blosum_score =float((row.split())[(Blosum[0].split()).index(aa2)])
            except ValueError as ve:
                print aa1,aa2,"Value Error",ve
                sys.exit(1)

    return blosum_score


def percent_id(n1,p1,n2,p2):

    """
        Get amino acid sequence P1 and amino acid sequence 2 and return
        the percentage id. In this implementation gaps are not counted
        in the length of the alignment
    """
    aa_match = 0
    ni = min(len(p1),len(p2))
    gaps =0
    for i1 in range(0,ni):
        if p1[i1] == p2[i1]:
            aa_match+=1
        elif '.' in p1[i1]+p2[i1]:
            gaps += 1
    identity = 100*aa_match/float(max(n1,n2))

    return (aa_match, identity, gaps)


def sim(a, b):

    """
        Find the similarity of the omega values a and b.
        This is the absolute difference between omega 1 and omega 2.
        Change from the the ver 1 where there was a  max on the differences
        between the two omega values
    """

    #To be done test this function thoroughly
    if  ('-' == a or '-' == b):return 0
    #account the insertion of gaps in the calculation of similarity score
    #This function returns zero if a gap has been inserted
    elif (a >= 1 or b >=  1):
        sim_ab = 1 -(float(abs(a - b))/max(a,b))
    else:
        sim_ab = 1 - abs(a - b)

    return sim_ab


def blosum_fire(blosum, a, b, K=5.63):

    """
```

```
        Get the blosum score and omega values a and b.
        Calculate similarity of the omega values a and b and use this
        as the selective pressure on the amino acids to scale the blosum
    """
    #Calculate the similariy of the omega values, The FIRE principle
    sim_ab = sim(a,b)
    #Calculate  the selective pressure on the using omega values
    sel_ab = math.exp(-(a+b))
    #Scaling the blosum score according to the selective pressure on the amino acids
    selection = blosum * sel_ab
    #Proportionality constant used to make Blosum values comparable to fire score
    #The K is basically the mean of the positive blosum scores for BLOSUM62
    similarity = K * (sim_ab) * (1 - sel_ab)
    Fire = selection + similarity

    return Fire


def do_align(p1,p2, data1, data2, Blosum_matrix,gop,gep):

    """
        use the dat in data1 and data2 to produce alignments and insert gaps where required
        This is the Modified Needleman and Wunsch Algorithm for dynamic programming
    """

    n1 = len(p1)
    n2 = len(p2)
    (up,diag,left)=(0,1,2)

    dmat = [[[0]*(n2+1) for i in range(n1+1)] for j in range(3)]
    direct  = [[['D']*(n2+1) for i in range(n1+1)] for j in range(3)]

    for i1 in range(1,n1+1):
        dmat[up][i1][0]=dmat[diag][i1][0]=dmat[left][i1][0]=(i1-1)*gep+gop
        direct[up][i1][0]=direct[diag][i1][0]=direct[left][i1][0]='U'
    for i2 in range(1,n2+1):
        dmat[up][0][i2]=dmat[diag][0][i2]=dmat[left][0][i2]=(i2-1)*gep+gop
        direct[up][0][i2]=direct[diag][0][i2]=direct[left][0][i2]='L'

    for i1 in range(0,n1):
        for i2 in range(0,n2):
            blosum = score_blosum(Blosum_matrix,p1[i1],p2[i2])
            Fire = blosum_fire(blosum,data1[i1][0],data2[i2][0])

            gop_U =  gop*math.exp(-(data1[i1][0]))
            gop_L =  gop*math.exp(-(data2[i2][0]))

            optD =  Fire + dmat[diag][i1][i2]
            optU =  gop_U + dmat[up][i1][i2+1]
            optL =  gop_L  + dmat[left][i1+1][i2]

            if optD > optL and optD > optU:
                direct[diag][i1+1][i2+1]='D'
                dmat[diag][i1+1][i2+1] = optD

            elif optL > optU:
                direct[diag][i1+1][i2+1]='L'
                dmat[diag][i1+1][i2+1] = optL

            else:
                direct[diag][i1+1][i2+1]='U'
                dmat[diag][i1+1][i2+1] = optU

           #level 0 -- up

            gep_U = gep*math.exp(-(data1[i1][0]))

            optU = gep_U + dmat[up][i1][i2+1]

            if optU > optD:
```

```
                direct[up][i1+1][i2+1] ='U'
                dmat[up][i1+1][i2+1] = optU

            else:
                direct[up][i1+1][i2+1] ='D'
                dmat[up][i1+1][i2+1] = optD

            #level 2 -- up

            gep_L =  gep*math.exp(-(data2[i2][0]))

            optL = gep_L + dmat[left][i1+1][i2]

            if optL > optD:
                direct[left][i1+1][i2+1]='L'
                dmat[left][i1+1][i2+1] = optL

            else:
                direct[left][i1+1][i2+1]='D'
                dmat[left][i1+1][i2+1] = optD


    i1=n1
    i2=n2
    s1=s2=[]
    level=diag

    while i1>=1 or i2>=1:
        if direct[level][i1][i2] in ['D','U']:
            s1=[data1[i1-1]]+s1
            i1d=i1-1
        else:
            i1d=i1
            s1=[['-']]+s1

        if direct[level][i1][i2] in ['D','L']:
            s2=[data2[i2-1]]+s2
            i2=i2-1

        else:
            s2=[['-']]+s2
        i1=i1d
        if direct[level][i1][i2] in ['D']: level = diag
        if direct[level][i1][i2] in ['U']: level = up
        if direct[level][i1][i2] in ['L']: level = left

    return (dmat, direct, s1,s2)


def trace_back(n1,n2,direct,p1,p2,options):

    """
       Determine the best alignment using the direction determined by the the Needleman
       and  Wunch algorithm in the "do_align" method.

    """
    i = n1
    j = n2
    scan = ''
    level = 1
    (up, diag, left) = (0, 1, 2)
    align_p1 = ""
    align_p2 = ""
    while (i> 0 or j>0):
       if direct[level][i][j]=='D':
            align_p1 = p1[i-1] + align_p1
            align_p2=p2[j-1] + align_p2
            level=diag
            scan = '.'+scan
            i=i-1
```

```
            j=j-1

        elif direct[level][i][j]=='L':
            align_p1 = "."+ align_p1
            align_p2=p2[j-1] + align_p2
            level=left
            scan = '<'+scan
            if options.seqalign:
                orig_aligns[0].insert(i,".")
            j=j-1

        elif direct[level][i][j]=='U':
            align_p1 = p1[i-1] + align_p1
            align_p2= "." + align_p2

            level = up
            scan = '>'+scan
            if options.seqalign:
                orig_aligns[1].insert(j,".")
            i=i-1

     return (scan,align_p1,align_p2)


def doRandom(fname1, fname2, p1, p2, data1, data2, Blosum_matrix, gop, gep, Gnuplot, g):

    """
       Generate statistics for the dN/dS values in the different values

    """
    n1 = len(p1)
    n2 = len(p2)
    import scipy
    show = lambda x:sys.stdout.write("%s: dN/dS Ave %5.2f, standard deviation  is %5.2f\n"%\
    (x[0],scipy.average(x[1]),scipy.std(x[1])))
    scores = []
    dmat, direct, s1,s2 = do_align(p1, p2, data1,data2,Blosum_matrix,gop,gep)
    actual = get_windows(s1, s2, n1, n2,Gnuplot,g)
    seq_data = [(fname1,data1), (fname2,data2)]
    map(show, seq_data)
    get = lambda d:d[0]
    s_1, s_2 = map(get, s1), map(get, s2)
    rm_str = lambda x : 0 if isinstance(x, str) else x
    s_1, s_2 = map(rm_str, s_1), map(rm_str, s_2)
    print "dN/dS Correlation is %5.3f"%(scipy.corrcoef(s_1,s_2)[0][1])
    for i in range(int(options.random)):
        random.shuffle(data2)
        dmat, direct, s1,s2 = do_align(p1, p2, data1,data2,Blosum_matrix,gop,gep)
        sim_score = get_windows(s1, s2, n1, n2,Gnuplot,g)
        scores.append(sim_score)
    ave = scipy.average(scores)
    dev = scipy.std(scores)
    print  "Actual dN/ds score (Old FIRE)=%5.2f."\
    "Randomised trials: average %4.1f stdev=%5.2f\n"%(actual,ave,dev)


def doPerturb():
    delta = float(options.perturb)
    for i in range(10):
        data2 = copy.deepcopy(data1)
        for i in range(n2):
            for j in range(len(data2[i])):
                data2[i][j] = random.uniform(data2[i][j]-delta,data2[i][j]+delta)
        dmat = [[0]*(n2+1) for i in range(n1+1)]
        direct  = [['D']*(n2+1) for i in range(n1+1)]
        do_align(data1, data2, dmat, direct)
        print dmat[n1][n2]


def get_windows(s1, s2, n1, n2,Gnuplot,g):
```

```
    """
       Get sequence s1 and sequence s2 and the lenght of the window.
       This function generaes the histograms. This initiates the
       varialbles for the plots.
    """
    sum = [0]
    s = 0
    sim_scores = 0
    histo = [0]*11
    N = len(s1)
    #print N
    for i in range(N):
        d = sim(s1[i][0],s2[i][0])
        sim_scores+=d
        sum.append(sim_scores)
        histo[int(d*10)]=histo[int(d*10)]+1
    histo[9]=histo[9]+histo[10]
    fire_sim =  (float(sim_scores)/(N))
    if options.histo:
        for i in range(10):
            print "%3d%%-%3d%%: %4d"%(i*10,(i+1)*10,histo[i])
    if not options.gnuplot:
        return fire_sim

    curr_vals = []
    window_lens = options.window_len
    window_lens = map(int,window_lens.split(","))
    for b in window_lens:
        curr = []
        for i in range(b/2,N-b/2):
            d = sum[i+b/2]-sum[i-b/2]
            curr.append([i,100*d/b])
        curr_vals.append(curr)
    colours = [3,4,5,4,6,5,2,0,8,9,10,11,12,13,14]
    g('set yrange [0:105]')
    if window_lens[0]<=4:
        line_type = "points lc "
    else:
        line_type = "lines lc "

    terminals = [ copy.deepcopy(g) for i in range(len(window_lens)) ]
    for i in range(0,len(window_lens)):
        g = terminals[i]
        if curr_vals[i]:
            plotargs = {"with":"lines lc {}".format(colours[i]),\
            "title":"Window length %d"%(window_lens[i]),'inline':True}
            plotd = Gnuplot.Data(curr_vals[i],**plotargs)
            g.plot(plotd,"60 title '60% Similarity threshold'")
            if not options.plot_fname:
               g('pause 5')

    return fire_sim


def InitGnuPlot(Gnuplot,fname1,fname2):

     """
         Instatitiate a gnuplot instance with labels for the x-axis and y-axis.

     """
     g = Gnuplot.Gnuplot(debug=1)
     g.title('Plot of omega similarity versus codon position: %s vs %s'%(fname1,fname2))
     g('set ylabel "Percentage similarity"')
     g('set xlabel "Codon position"')
     g('set key bottom left')
     if options.plot_fname:
         g("set terminal {}".format(options.gnu_term))
         g("set output '{}'".format(options.plot_fname))
     return g
```

```
def seq_align(args):
    from Bio import Seq
    from Bio.Align.Generic import Alignment
    from Bio.Align import MultipleSeqAlignment
    from Bio.SeqRecord import SeqRecord
    from Bio.Alphabet import IUPAC
    from Bio import AlignIO
    (p1,p2) = str(orig_aligns[0]), str(orig_aligns[1])
    from Bio.Seq import Seq
    seq1 = MultipleSeqAlignment([SeqRecord(Seq(p1, IUPAC.protein), id= args[0], description=''),
                                 SeqRecord(Seq(p2, IUPAC.protein), id= args[1], description='')])
    if options.seqout:

        out = open(options.seqout,'w')
    else:
        out = sys.stdout

    AlignIO.write(seq1, out, options.seqformat)


def main():


    parser = InitStuff()
    global options
    options, args = parser.parse_args()
    regexp = {'m2': "\A *\d+ ([A-Z]).*(\d+\.\d+) *\+-",
              'm3':"\A *\d+ ([A-Z]).*\) +(\d+\.\d+)",
              '1':"\A *\d+ ([A-Z]).*\) +(\d+\.\d+)"}

    try:
        regexp = regexp[options.regexp]
    except KeyError:
        print 'invalid model parameters ', regexp[options.regexp]
        parser.print_help()
    fname1,fname2 = get_filenames(args,parser)
    p1, data1 = get_data(fname1,regexp)
    p2, data2 = get_data(fname2,regexp)
    n1 = len(data1)
    n2 = len(data2)
    matrix = int(options.blosum)

    if options.seqout:
        options.seqalign = True
    if options.gnuplot:
        import Gnuplot
        g = InitGnuPlot(Gnuplot,fname1,fname2)
    else:
        Gnuplot= None
        g = None
    if options.seqalign:
        from Bio import Seq
        global orig_aligns
        orig_aligns = [Seq.MutableSeq(p1), Seq.MutableSeq(p2)]

    Blosum_matrix, Blosum_value,gop,gep = Run_Blosum(matrix)
    if options.gop:
        gop = float(options.gop)
    if options.gep:
        gep = float(options.gep)

    (dmat, direct, s1,s2) = do_align(p1, p2, data1,data2,Blosum_matrix,gop,gep)
    (path,align_p1, align_p2) = trace_back(n1, n2, direct,p1,p2,options)
    (aa_match, identity, gaps) = percent_id(n1,align_p1, n2,align_p2)
    fire_sim = get_windows(s1,s2, n1 , n2,Gnuplot,g)
    Fire_score = (fire_sim + identity)/2
    if not options.quiet:
        print 'BLOSUM-FIRE Score: {0:.2f}'.format(float(Fire_score))
```

```
        print 'Percent Identity score: {0:.2f}'.format(identity)
        print 'Residue Matches: {0}/{1}'.format(aa_match, max(len(align_p1),len(align_p2)))
        print 'dN/dS similarity score (Old  FIRE): {0:.2f} \n\n'.format(fire_sim)
    if options.seqalign:
        seq_align(args)
    if options.random:
        doRandom(fname1, fname2, p1, p2, data1, data2, Blosum_matrix, gop, gep, Gnuplot, g)

if __name__ == '__main__':
        main()
```

Appendix B: The BLOSUM matrix substitution file

The BLOSUM substitution matrices in the *Blosum_matrix.py* file are used to score aligned residues.

```
#  Blosum_matrix.py                                                            #
#  matrices for scoring aligned amino acids                                    #
#  This script is called by the FIRE algorithm                                 #
#                                                                              #
#                                                                              #
Blosum30=("""~ A  R  N  D  C  Q  E  G  H  I  L  K  M  F  P  S  T  W  Y  V  B  Z  X  *
A  4 -1  0  0 -3  1  0  0 -2  0 -1  0  1 -2 -1  1  1 -5 -4  1  0  0  0 -7
R -1  8 -2 -1 -2  3 -1 -2 -1 -3 -2  1  0 -1 -1 -1 -3  0  0 -1 -2  0 -1 -7
N  0 -2  8  1 -1 -1 -1  0 -1  0 -2  0  0 -1 -3  0  1 -7 -4 -2  4 -1  0 -7
D  0 -1  1  9 -3 -1  1 -1 -2 -4 -1  0 -3 -5 -1  0 -1 -4 -1 -2  5  0 -1 -7
C -3 -2 -1 -3 17 -2  1 -4 -5 -2  0 -3 -2 -3 -3 -2 -2 -2 -6 -2 -2  0 -2 -7
Q  1  3 -1 -1 -2  8  2 -2  0 -2 -2  0 -1 -3  0 -1  0 -1 -1 -3 -1  4  0 -7
E  0 -1 -1  1  1  2  6 -2  0 -3 -1  2 -1 -4  1  0 -2 -1 -2 -3  0  5 -1 -7
G  0 -2  0 -1 -4 -2 -2  8 -3 -1 -2 -1 -2 -3 -1  0 -2  1 -3 -3  0 -2 -1 -7
H -2 -1 -1 -2 -5  0  0 -3 14 -2 -1 -2  2 -3  1 -1 -2 -5  0 -3 -2  0 -1 -7
I  0 -3  0 -4 -2 -2 -3 -1 -2  6  2 -2  1  0 -3 -1  0 -3 -1  4 -2 -3  0 -7
L -1 -2 -2 -1  0 -2 -1 -2 -1  2  4 -2  2  2 -3 -2  0 -2  3  1 -1 -1  0 -7
K  0  1  0  0 -3  0  2 -1 -2 -2 -2  4  2 -1  1  0 -1 -2 -1 -2  0  1  0 -7
M  1  0  0 -3 -2 -1 -1 -2  2  1  2  2  6 -2 -4 -2  0 -3 -1  0 -2 -1  0 -7
F -2 -1 -1 -5 -3 -3 -4 -3 -3  0  2 -1 -2 10 -4 -1 -2  1  3  1 -3 -4 -1 -7
P -1 -1 -3 -1 -3  0  1 -1  1 -3 -3  1 -4 -4 11 -1  0 -3 -2 -4 -2  0 -1 -7
S  1 -1  0  0 -2 -1  0  0 -1 -1 -2  0 -2 -1 -1  4  2 -3 -2 -1  0 -1  0 -7
T  1 -3  1 -1 -2  0 -2 -2 -2  0  0 -1  0 -2  0  2  5 -5 -1  1  0 -1  0 -7
W -5  0 -7 -4 -2 -1 -1  1 -5 -3 -2 -2 -3  1 -3 -3 -5 20  5 -3 -5 -1 -2 -7
Y -4  0 -4 -1 -6 -1 -2 -3  0 -1  3 -1 -1  3 -2 -2 -1  5  9  1 -3 -2 -1 -7
V  1 -1 -2 -2 -2 -3 -3 -3 -3  4  1 -2  0  1 -4 -1  1 -3  1  5 -2 -3  0 -7
B  0 -2  4  5 -2 -1  0  0 -2 -2 -1  0 -2 -3 -2  0  0 -5 -3 -2  5  0 -1 -7
Z  0  0 -1  0  0  4  5 -2  0 -3 -1  1 -1 -4  0 -1 -1 -1 -2 -3  0  4  0 -7
X  0 -1  0 -1 -2  0 -1 -1 -1  0  0  0  0 -1 -1  0  0 -2 -1  0 -1  0 -1 -7
* -7 -7 -7 -7 -7 -7 -7 -7 -7 -7 -7 -7 -7 -7 -7 -7 -7 -7 -7 -7 -7 -7  1""")

Blosum45=("""~ A  R  N  D  C  Q  E  G  H  I  L  K  M  F  P  S  T  W  Y  V  B  Z  X  *
A  5 -2 -1 -2 -1 -1 -1  0 -2 -1 -1 -1 -1 -2 -1  1  0 -2 -2  0 -1 -1  0 -5
R -2  7  0 -1 -3  1  0 -2  0 -3 -2  3 -1 -2 -2 -1 -1 -2 -1 -2 -1  0 -1 -5
N -1  0  6  2 -2  0  0  0  1 -2 -3  0 -2 -2 -2  1  0 -4 -2 -3  4  0 -1 -5
D -2 -1  2  7 -3  0  2 -1  0 -4 -3  0 -3 -4 -1  0 -1 -4 -2 -3  5  1 -1 -5
C -1 -3 -2 -3 12 -3 -3 -3 -3 -3 -2 -3 -2 -2 -4 -1 -1 -5 -3 -1 -2 -3 -2 -5
Q -1  1  0  0 -3  6  2 -2  1 -2 -2  1  0 -4 -1  0 -1 -2 -1 -3  0  4 -1 -5
E -1  0  0  2 -3  2  6 -2  0 -3 -2  1 -2 -3  0  0 -1 -3 -2 -3  1  4 -1 -5
G  0 -2  0 -1 -3 -2 -2  7 -2 -4 -3 -2 -2 -3 -2  0 -2 -2 -3 -3 -1 -2 -1 -5
H -2  0  1  0 -3  1  0 -2 10 -3 -2 -1  0 -2 -2 -1 -2 -3  2 -3  0  0 -1 -5
I -1 -3 -2 -4 -3 -2 -3 -4 -3  5  2 -3  2  0 -2 -2 -1 -2  0  3 -3 -3 -1 -5
L -1 -2 -3 -3 -2 -2 -2 -3 -2  2  5 -3  2  1 -3 -3 -1 -2  0  1 -3 -2 -1 -5
K -1  3  0  0 -3  1  1 -2 -1 -3 -3  5 -1 -3 -1 -1 -1 -2 -1 -2  0  1 -1 -5
M -1 -1 -2 -3 -2  0 -2 -2  0  2  2 -1  6  0 -2 -2 -1 -2  0  1 -2 -1 -1 -5
F -2 -2 -2 -4 -2 -4 -3 -3 -2  0  1 -3  0  8 -3 -2 -1  1  3  0 -3 -3 -1 -5
P -1 -2 -2 -1 -4 -1  0 -2 -2 -2 -3 -1 -2 -3  9 -1 -1 -3 -3 -3 -2 -1 -1 -5
S  1 -1  1  0 -1  0  0  0 -1 -2 -3 -1 -2 -2 -1  4  2 -4 -2 -1  0  0  0 -5
T  0 -1  0 -1 -1 -1 -1 -2 -2 -1 -1 -1 -1 -1 -1  2  5 -3 -1  0  0 -1  0 -5
```

```
W -2 -2 -4 -4 -5 -2 -3 -2 -3 -2 -2 -2 -2  1 -3 -4 -3 15  3 -3 -4 -2 -2 -5
Y -2 -1 -2 -2 -3 -1 -2 -3  2  0  0 -1  0  3 -3 -2 -1  3  8 -1 -2 -2 -1 -5
V  0 -2 -3 -3 -1 -3 -3 -3 -3  3  1 -2  1  0 -3 -1  0 -3 -1  5 -3 -3 -1 -5
B -1 -1  4  5 -2  0  1 -1  0 -3 -3  0 -2 -3 -2  0  0 -4 -2 -3  4  2 -1 -5
Z -1  0  0  1 -3  4  4 -2  0 -3 -2  1 -1 -3 -1  0 -1 -2 -2 -3  2  4 -1 -5
X  0 -1 -1 -1 -2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  0  0 -2 -1 -1 -1 -1 -1 -5
* -5 -5 -5 -5 -5 -5 -5 -5 -5 -5 -5 -5 -5 -5 -5 -5 -5 -5 -5 -5 -5 -5  1""")

Blosum62=("""~ A  R  N  D  C  Q  E  G  H  I  L  K  M  F  P  S  T  W  Y  V  B  Z  X  *
A  4 -1 -2 -2  0 -1 -1  0 -2 -1 -1 -1 -1 -2 -1  1  0 -3 -2  0 -2 -1  0 -4
R -1  5  0 -2 -3  1  0 -2  0 -3 -2  2 -1 -3 -2 -1 -1 -3 -2 -3 -1  0 -1 -4
N -2  0  6  1 -3  0  0  0  1 -3 -3  0 -2 -3 -2  1  0 -4 -2 -3  3  0 -1 -4
D -2 -2  1  6 -3  0  2 -1 -1 -3 -4 -1 -3 -3 -1  0 -1 -4 -3 -3  4  1 -1 -4
C  0 -3 -3 -3  9 -3 -4 -3 -3 -1 -1 -3 -1 -2 -3 -1 -1 -2 -2 -1 -3 -3 -2 -4
Q -1  1  0  0 -3  5  2 -2  0 -3 -2  1  0 -3 -1  0 -1 -2 -1 -2  0  3 -1 -4
E -1  0  0  2 -4  2  5 -2  0 -3 -3  1 -2 -3 -1  0 -1 -3 -2 -2  1  4 -1 -4
G  0 -2  0 -1 -3 -2 -2  6 -2 -4 -4 -2 -3 -3 -2  0 -2 -2 -3 -3 -1 -2 -1 -4
H -2  0  1 -1 -3  0  0 -2  8 -3 -3 -1 -2 -1 -2 -1 -2 -2  2 -3  0  0 -1 -4
I -1 -3 -3 -3 -1 -3 -3 -4 -3  4  2 -3  1  0 -3 -2 -1 -3 -1  3 -3 -3 -1 -4
L -1 -2 -3 -4 -1 -2 -3 -4 -3  2  4 -2  2  0 -3 -2 -1 -2 -1  1 -4 -3 -1 -4
K -1  2  0 -1 -3  1  1 -2 -1 -3 -2  5 -1 -3 -1  0 -1 -3 -2 -2  0  1 -1 -4
M -1 -1 -2 -3 -1  0 -2 -3 -2  1  2 -1  5  0 -2 -1 -1 -1 -1  1 -3 -1 -1 -4
F -2 -3 -3 -3 -2 -3 -3 -3 -1  0  0 -3  0  6 -4 -2 -2  1  3 -1 -3 -3 -1 -4
P -1 -2 -2 -1 -3 -1 -1 -2 -2 -3 -3 -1 -2 -4  7 -1 -1 -4 -3 -2 -2 -1 -2 -4
S  1 -1  1  0 -1  0  0  0 -1 -2 -2  0 -1 -2 -1  4  1 -3 -2 -2  0  0  0 -4
T  0 -1  0 -1 -1 -1 -1 -2 -2 -1 -1 -1 -1 -2 -1  1  5 -2 -2  0 -1 -1  0 -4
W -3 -3 -4 -4 -2 -2 -3 -2 -2 -3 -2 -3 -1  1 -4 -3 -2 11  2 -3 -4 -3 -2 -4
Y -2 -2 -2 -3 -2 -1 -2 -3  2 -1 -1 -2 -1  3 -3 -2 -2  2  7 -1 -3 -2 -1 -4
V  0 -3 -3 -3 -1 -2 -2 -3 -3  3  1 -2  1 -1 -2 -2  0 -3 -1  4 -3 -2 -1 -4
B -2 -1  3  4 -3  0  1 -1  0 -3 -4  0 -3 -3 -2  0 -1 -4 -3 -3  4  1 -1 -4
Z -1  0  0  1 -3  3  4 -2  0 -3 -3  1 -1 -3 -1  0 -1 -3 -2 -2  1  4 -1 -4
X  0 -1 -1 -1 -2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -2  0  0 -2 -1 -1 -1 -1 -1 -4
* -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4  1""")


Blosum80=("""~ A  R  N  D  C  Q  E  G  H  I  L  K  M  F  P  S  T  W  Y  V  B  Z  X  *
A  7 -3 -3 -3 -1 -2 -2  0 -3 -3 -3 -1 -2 -4 -1  2  0 -5 -4 -1 -3 -2 -1 -8
R -3  9 -1 -3 -6  1 -1 -4  0 -5 -4  3 -3 -5 -3 -2 -2 -5 -4 -4 -2  0 -2 -8
N -3 -1  9  2 -5  0 -1 -1  1 -6 -6  0 -4 -6 -4  1  0 -7 -4 -5  5 -1 -2 -8
D -3 -3  2 10 -7 -1  2 -3 -2 -7 -7 -2 -6 -6 -3 -1 -2 -8 -6 -6  6  1 -3 -8
C -1 -6 -5 -7 13 -5 -7 -6 -7 -2 -3 -6 -3 -4 -6 -2 -2 -5 -5 -2 -6 -7 -4 -8
Q -2  1  0 -1 -5  9  3 -4  1 -5 -4  2 -1 -5 -3 -1 -1 -4 -3 -4 -1  5 -2 -8
E -2 -1 -1  2 -7  3  8 -4  0 -6 -6  1 -4 -6 -2 -1 -2 -6 -5 -4  1  6 -2 -8
G  0 -4 -1 -3 -6 -4 -4  9 -4 -7 -7 -3 -5 -6 -5 -1 -3 -6 -6 -6 -2 -4 -3 -8
H -3  0  1 -2 -7  1  0 -4 12 -6 -5 -1 -4 -2 -4 -2 -3 -4  3 -5 -1  0 -2 -8
I -3 -5 -6 -7 -2 -5 -6 -7 -6  7  2 -5  2 -1 -5 -4 -2 -5 -3  4 -6 -6 -2 -8
L -3 -4 -6 -7 -3 -4 -6 -7 -5  2  6 -4  3  0 -5 -4 -3 -4 -2  1 -7 -5 -2 -8
K -1  3  0 -2 -6  2  1 -3 -1 -5 -4  8 -3 -5 -2 -1 -1 -6 -4 -4 -1  1 -2 -8
M -2 -3 -4 -6 -3 -1 -4 -5 -4  2  3 -3  9  0 -4 -3 -1 -3 -3  1 -5 -3 -2 -8
F -4 -5 -6 -6 -4 -5 -6 -6 -2 -1  0 -5  0 10 -6 -4 -4  0  4 -2 -6 -6 -3 -8
P -1 -3 -4 -3 -6 -3 -2 -5 -4 -5 -5 -2 -4 -6 12 -2 -3 -7 -6 -4 -4 -2 -3 -8
S  2 -2  1 -1 -2 -1 -1 -1 -2 -4 -4 -1 -3 -4 -2  7  2 -6 -3 -3  0 -1 -1 -8
T  0 -2  0 -2 -2 -1 -2 -3 -3 -2 -3 -1 -1 -4 -3  2  8 -5 -3  0 -1 -2 -1 -8
W -5 -5 -7 -8 -5 -4 -6 -6 -4 -5 -4 -6 -3  0 -7 -6 -5 16  3 -5 -8 -5 -5 -8
Y -4 -4 -4 -6 -5 -3 -5 -6  3 -3 -2 -4 -3  4 -6 -3 -3  3 11 -3 -5 -4 -3 -8
V -1 -4 -5 -6 -2 -4 -4 -6 -5  4  1 -4  1 -2 -4 -3  0 -5 -3  7 -6 -4 -2 -8
B -3 -2  5  6 -6 -1  1 -2 -1 -6 -7 -1 -5 -6 -4  0 -1 -8 -5 -6  6  0 -3 -8
Z -2  0 -1  1 -7  5  6 -4  0 -6 -5  1 -3 -6 -2 -1 -2 -5 -4 -4  0  6 -1 -8
X -1 -2 -2 -3 -4 -2 -2 -3 -2 -2 -2 -2 -3 -3 -1 -1 -5 -3 -2 -3 -1 -2 -8
* -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8  1""")

def do_blosum(blosum):

    if blosum < 30:
        Blosum=Blosum30
        B_var="Blosum30"
        gop=-3
        gep=-1

    elif blosum >= 30 and blosum < 60:
```

```
        Blosum=Blosum45
        B_var="Blosum45"
        gop=-7
        gep=-1

    elif blosum >= 60 and blosum < 80:
        Blosum=Blosum62
        B_var="Blosum62"
        gop=-4
        gep=-1

    elif blosum >= 80:
        Blosum=Blosum80
        B_var="Blosum80"
        gop=-16
        gep=-4

    return (Blosum,B_var,gop,gep)
```

Appendix C: The *dblookup* utility

The *dblookup* utility is a pipeline that compiled the EvoDB database. The utility accesses local databases using EMBOSS suite of programs.

```
import sys
import subprocess
import pickle
import shutil
import time
import datetime
import re
import os


def init_dir(init_dir, filename):

    """
        get the filenamem, check the working directory with filename as name.
        if directory does not exist exit
    """

    if verbose:print'\nCHECK_DIR\n'
    new_dir = os.path.join(init_dir,'FIRE',filename)
    if verbose:print os.getcwd()
    if not os.path.exists(new_dir):
        os.makedirs(new_dir)
    dest = os.path.join(new_dir, filename)
    os.chdir(new_dir)
    if verbose:print os.getcwd()
    if not os.path.exists(dest):
        sys.exit(1)
    return new_dir


def url_get(dbName, format, entryId):

    """
        Use url to fecth sequncea  information this uses the shell utility
        Wget as urllib mmodules raise errors on the cluster machine.
        The data is downloaded to stdout
    """


    baseUrl = 'http://www.ebi.ac.uk/Tools/dbfetch/dbfetch'
    url = baseUrl + '/' + dbName + '/' + entryId
    if format != None:
        url += '/' + format

    result = os.popen("wget -O - -o /dev/null %s"%url).read()

    return result


def pfam_parser(filename):

    """
       open the pfam file and extract the information in the file in the form of a dict
    """
```

```
    if verbose:print'\nPFAM_PARSER\n'
    pfam_xref = {}
    pfam_xref[filename]= []
    try:
        with open(filename) as fp:
            info = fp.readlines()
        for line in info:
            found_acc = re.match(find_acc, line)
            if found_acc:
                pfam_xref[filename].append([found_acc.group(1)+found_acc.group(2),\
                found_acc.group(3)])
    except IOError:
        with open(filename+'.psr_fail','w') as fp2:
            fp2.write('NOT FOUND')
            sys.exit(0)

    return pfam_xref


def db_fetch(pfam_xref):
    verbose = True
    """
        Get the pfam cross references,
        -fetch the reference proteins using either entret for a local database or url
        -open the protein file to get the reference information
         for the original nucleotide sequences.
        -use the reference information to get the nucleotide information from a
         local database or using url.
        -use the emboss utility extractrfeat to extract the nucleotide
        -use the emboss utility tranalign to check if the correctness of the extracted
         sequences.
        -if sequnces cannot be found add them to missing list which will be written to file
         return a dict of all the found sequnces named data and another not_found dict that
         contains all the sequences which could not be found
    """

    pfam_id = pfam_xref.keys()[0]
    prot_ids = pfam_xref.values()[0]
    not_found = {}
    not_found[pfam_id] = []
    data = {}
    data['Pfam acc'] = pfam_id

    for xref in prot_ids:
        acc_id = xref[1]
        swiss_id = xref[0]
        if verbose: print xref
        args = 'entret  -sequence swissprot:%s -outfile stdout'%(acc_id)
        p = subprocess.Popen(args, shell= True, stdout =subprocess.PIPE, stderr=subprocess.PIPE)
        (uniprot_seq, stderrdata) = p.communicate()
        data[swiss_id]=[acc_id]
        if not uniprot_seq:
            args = 'entret  -sequence trembl:%s -outfile stdout'%(acc_id)
            p = subprocess.Popen(args, shell= True, stdout =subprocess.PIPE, stderr=subprocess.PIPE)
            (uniprot_seq, stderrdata) = p.communicate()
        if not uniprot_seq:
            uniprot_seq = url_get('uniprotKB','uniprot', acc_id)
        if 'ERROR' in uniprot_seq:
            not_found[pfam_id].append(swiss_id+'>swiss')
            continue
        elif not uniprot_seq:
             not_found[pfam_id].append(swiss_id+'>swiss')
             continue
        with open(acc_id,'w') as fp1:
            fp1.write(uniprot_seq)
        if verbose: print uniprot_seq
        dbxref = re.compile("DR   EMBL; (\w*);\s*(\w*.\w+);.*")
        found_dbxref = re.findall(dbxref, uniprot_seq)
        nuc_seq = None
        align_out = None
```

```
        if found_dbxref:
            counter = 1
            found = len(found_dbxref)
            for xref in found_dbxref:
                (gene_id, prot_id) = xref
                if verbose:print len(found_dbxref),xref
                if verbose:print '\nENTRET GENBANK\n'
                args = 'entret  -sequence genbank:%s -outfile stdout'%(gene_id)
                p1 = subprocess.Popen(args, shell= True, stdout =subprocess.PIPE, stderr=subprocess.
                (nuc_seq, stderrdata) = p1.communicate()
                if not nuc_seq:
                    if verbose:print '\nEMBL GET\n'
                    nuc_seq = url_get('embl','default',gene_id)
                if not nuc_seq:
                    if counter == found:
                        not_found[pfam_id].append(swiss_id+'>gene_id')
                        break
                    else:
                        continue
                if verbose:print nuc_seq
                with open(gene_id,'w') as fp2:
                    fp2.write(nuc_seq)
                if verbose:print '\nEXTRACTFEAT\n'
                args='extractfeat -sequence %s -outseq stdout'\
                ' -type CDS -value %s -join -sid1 %s'%(gene_id,\
                prot_id, swiss_id)
                p1 = subprocess.Popen(args, \
                shell= True, stdout =subprocess.PIPE, stderr=subprocess.PIPE)
                (nuc_output, stderrdata) = p1.communicate()
                if not nuc_output:
                     if counter == found:
                         not_found[pfam_id].append(swiss_id+'>exft')
                         break
                     else:
                         continue
                if verbose:print nuc_output
                with open(gene_id,'w') as fp3:
                    fp3.write(nuc_output)

                if verbose:print '\nTRANALIGN >>\n'
                args='tranalign -asequence %s -bsequence %s -outseq'\
                'stdout -table 0 -osformat fasta -auto'%(\
                gene_id, acc_id)
                p = subprocess.Popen(args, shell= True,\
                stdout =subprocess.PIPE, stderr=subprocess.PIPE)
                (align_out, stderrdata) = p.communicate()
                if align_out:
                    if verbose: print gene_id, prot_id,'\n',align_out
                    nuc_file = swiss_id.replace('/','_')
                    if verbose:print '\nTRANALIGN <<\n'
                    with open(nuc_file,'w') as fp3:
                        fp3.write(align_out)
                        data[swiss_id].append(gene_id)
                        data[swiss_id].append(prot_id)
                        break
                counter += 1
        if not found_dbxref:
            not_found[pfam_id].append(swiss_id+'>xref')
            continue
        if not align_out:
            not_found[pfam_id].append(swiss_id+'>tran')
            continue
    if verbose:print '\nDATA\n',data,'\nNOT FOUND\n',not_found
    return data, not_found


def rm_missing(data, not_found):

    """
        using data and not_found to  remove  missing proteins from original pfam file
```

```
        return the filename of the nucleotide file  and protein file
    """

    if verbose:print'\nRM_MISSING\n',os.getcwd()

    acc_ids = ' '.join([prot_id for prot_id in  data.keys()])

    pfam_id = data['Pfam acc']
    try:
        with open(pfam_id) as fp1:
            pfam_data = fp1.readlines()
    except:
        pass

    find_seq = re.compile("(\w+)\/(\w+-\w+)\s+(\S+)")
    nuc_seq = []
    for line in pfam_data:
        found_DR = re.search(find_DR, line)
        if found_DR:
            prot_seq.append(line)
        else:
            found_seq = re.search(find_seq, line)
            if found_seq:
                seq_name = found_seq.group(1)+'/'+found_seq.group(2)
                regex = re.compile(seq_name)
                if re.search(regex, str(acc_ids)):
                    try:
                        seq_filename = seq_name.replace('/','_')
                        with open(seq_filename) as fp2:
                            nuc_data = fp2.read()
                            nuc_seq.append(nuc_data)
                            prot_seq.append(line)
                    except:
                        pass

    prot_file =  pfam_id+'.prot'
    with open(prot_file,'w') as fp3:
        fp3.writelines(prot_seq)

    nuc_file = pfam_id+'.nuc'
    with open(nuc_file,'w') as fp4:
        fp4.writelines(nuc_seq)
    return prot_file, nuc_file



def finalise_seq(prot_file, nuc_file, filename):

    """
        get the protein file name and the nucleoatide sequence files
        Run traanlign to get the nucleotide sequnces for the pfam files
    """

    if verbose:print '\nFINALISE SEQ\n'
    outseq = filename+'.nuc'
    args = 'tranalign -asequence %s -bsequence %s -outseq %s -osformat nexusnon'%(nuc_file,
    prot_file, outseq)
    p = subprocess.Popen(args, shell= True, stdout =subprocess.PIPE, stderr=subprocess.PIPE)
    (out, stderrdata) = p.communicate()
    if verbose:print '\nTRANALIGN OUT\n',out

    return outseq



def nexus_parser(outseq):

    if verbose:print '\nPHYLIP_PARSER\n'
    new_nexus = []
```

```
    seq_names = []
    seq_name  =  re.compile("(^\S{5,13}/)(\d+-\d+)")
    start = False
    with open(outseq) as fp:
        for line in fp:
           found_seq_name = re.match(seq_name, line)
           if found_seq_name:
               new_line = found_seq_name.group()+'\n'
               new_nexus.append(new_line)
               seq_names.append(found_seq_name.group())
           else:new_nexus.append(line)
    with open(outseq,'w') as fp2:
      fp2.writelines(new_nexus)

    return seq_names


def gen_trees(tree_dir, seq_names, not_found):
    if verbose:print"GEN TREES"
    from Bio import Phylo
    family_name = not_found.keys()[0]
    tree_file = ''.join([family_name.split('.')[0],'.tree'])
    original_tree = os.path.join(tree_dir,tree_file)
    with open(original_tree) as fp:
        tree_data = fp.read()

    fix_bracket = re.compile("\)(\d+.\d+):-*\d+.\d+")
    found = re.search(fix_bracket, tree_data)
    Pruned = []
    while found:
        tree_data = tree_data.replace(found.group(1),'')
        found = re.search(fix_bracket, tree_data)

    with open(tree_file,'w') as fp2:
        fp2.write(tree_data)

    tree =  Phylo.read(tree_file,'newick')
    tree_values = [ str(v) for v in tree.get_terminals() ]
    missing_values = [value for value in tree_values if value not in seq_names ]
    if verbose:print"Prunning ",len(missing_values)," from the tree"
    if verbose: print os.getcwd()
    if any(missing_values):
        for branch in missing_values:
            try:
               tree.prune(branch)
               if verbose:print "Pruned ",branch
               Pruned.append(branch)
            except:
               if verbose: print"Prune Failed",branch
               not_found[family_name].append(branch+'>prune_fail')
        with open(tree_file,'w') as fp2:
            Phylo.write(tree, fp2,'newick')

    return tree_file, not_found, Pruned


def run_codeml(bin_dir, outseq, tree_file):
    """
        get the filename of the sequence file and the filename of the tree file
        and run codeml
    """
    if verbose:print'\nCODEML\n',os.getcwd()
    codeml_exe = os.path.join(bin_dir,'codeml')

    data = """
      seqfile = %s
     treefile = %s
          outfile = mlc
        noisy = 0
      verbose = 0
```

```
      runmode = 0
      seqtype = 1
    CodonFreq = 2
        clock = 0
       aaDist = 0
        model = 0
      NSsites = 2
        icode = 0
        Mgene = 0
    fix_kappa = 0
    fix_omega = 0
        ncatG = 10
        getSE = 0
    RateAncestor = 0
    Small_Diff = .45e-6
    cleandata = 1
    fix_blength = 0\n"""

    with open('codeml.ctl','w') as fp:
        fp.writelines(data%(outseq, tree_file))

    p = subprocess.Popen(codeml_exe, shell= True, stdout =subprocess.PIPE,
    stderr=subprocess.PIPE)
    (stdoutdata, stderrdata) = p.communicate()
    if verbose:print'\nCODEML\n',stdoutdata,'\n',stderrdata


def get_rst():

    """
        Open the rst file and get the Bayes Empirical Bayes values with the MLEs
    """
    if verbose:print'\nGET_RST\n'
    locate = False
    start_line = 0
    find = re.compile("Bayes Empirical Bayes")
    try:
        with open('rst') as rst_data:
            raw = rst_data.readlines()
    except IOError as ioerror:
        print ioerror
        sys.exit(1)
    else:
        for line in raw:
            start_line+=1
            locate = find.match(line)
            if locate:
                return start_line
    if not locate:
        with open('fail_rst','w') as fp:
            fp.write("Failed to parse Bayes Empirical Bayes output\n")

    return start_line


def get_BEB(outseq, start_line):

    """
       Open the file and extract the Omega Values using the start line and write the information
       to seq.fire which will be used by FIRE algorithm
    """
    if verbose:print'\nGET_BEB\n'
    fire = outseq.replace('nuc','fire')
    check_line = re.compile("\[.\d+.*[a-zA-Z]")
    get_line = 0
    data = []
    with open('rst') as rst_data:
        raw = rst_data.readlines()
    for line in raw:
        search_term =str(line.split())
```

```
        get_line+=1

        if get_line > (start_line+2):

             if check_line.match(search_term):
                 data.append(line)
             else:
                 with open(fire,"w") as fp:
                     fp.writelines(data)

                 if verbose:
                     with open(fire) as rst:
                        print rst.read()
                 return fire
    if not any(data):
        with open(fire,"w") as fp:
             fp.writelines(data)
             return fire


def clean_up(start_time, data, not_found, Pruned):

    """
        Postprocessing Do some final cleanup and delete the files tc conserve space on
        the system. Write data to files  simple data of sequences worked on and data of
        those sequences not found and the trace of where these where not found
        The files deleted are not reauired. The codeml file is left
        behind to enable regeneration of the data
    """

    data_write  = [ items[0]+', '+', '.join(items[1])+'\n' for items in data.items()]
    end_time = time.time()
    time_taken_msge = 'Time taken : ' + str(datetime.timedelta(seconds = end_time - start_time))+'\n
    data_write.insert(len(data_write), time_taken_msge)
    pruned_write = [seq+'\n' for seq in Pruned ]

    data_filename = data['Pfam acc']+'.data'
    with open(data_filename,'w') as fd:
        fd.writelines(data_write)

    not_found_write = [ value+'\n' for value in not_found.values()[0]]
    not_found_filename = not_found.keys()[0]+'.ntfnd'
    with open(not_found_filename,'w') as fd:
        fd.writelines(not_found_write)

    pruned_filename =  data['Pfam acc']+'.pruned'
    with open(pruned_filename,'w') as fd:
        fd.writelines(pruned_write)

    if verbose:print'\nCLEAN_UP\n'
    for f in remove_list:
        try:
            os.remove(f)
        except:
            pass
    if verbose:print'+'*50


def main(filename):
    start_time = time.time()
    start_dir = os.getcwd()
    new_dir = init_dir(start_dir, filename)
    pfam_xref = pfam_parser(filename)
    data, not_found = db_fetch(pfam_xref)
    prot_file, nuc_file = rm_missing(data, not_found)
    outseq = finalise_seq(prot_file, nuc_file, filename)
    finalise_seq(prot_file, nuc_file, filename)
    seq_names = nexus_parser(outseq)
    tree_file, not_found, Pruned  = gen_trees(tree_dir,seq_names, not_found)
    run_codeml(bin_dir, outseq, tree_file)
```

```
    get_BEB(outseq, get_rst())
    clean_up(start_time, data, not_found, Pruned)


if __name__ == '__main__':
    verbose = True
    tree_dir = '/global/pfam/trees/seed'
    bin_dir = '/home/andhlovu/bin'
    main(sys.argv[1])
```