Prediction of Indel Flanking Regions and Its Application in the Alignment of Multiple Protein Sequences

Mufleh Saleh Al-Shatnawi

A Thesis
in
The Department
of
Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of Doctor of Philosophy at
Concordia University
Montréal, Québec, Canada

September 2015

Concordia University
SCHOOL OF GRADUATE STUDIES

This is to certify that the thesis prepared

By: **Mufleh Saleh Al-Shatnawi**

Entitled: **Prediction of Indel Flanking Regions and Its Application in the Alignment of Multiple Protein Sequences**

and submitted in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY (Electrical and Computer Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining commitee:

_____ Chair
Dr.C. Constantinides

_____ External Examiner
Dr. B. Ma

_____ External to Program
Dr. C.-Y. Su

_____ Examiner
Dr. C. Wang

_____ Examiner
Dr. W.-P. Zhu

_____ Thesis Co-Supervisor
Dr. M.O. Ahmad

_____ Thesis Co-Supervisor
Dr. M.N.S. Swamy

Approved by _____
Dr. A.R. Sebak, Graduate Program Director

September 10, 2015.

_____
Dr. A. Asif, Dean
Faculty of Engineering and Computer Science

Abstract

Prediction of Indel Flanking Regions and Its Application in the Alignment of Multiple Protein Sequences

Mufleh Saleh Al-Shatnawi, Ph.D.

Concordia University, 2015

Proteins are the most important molecules in living organism, and they are involved in every function of the cells, such as signal transmission, metabolic regulation, transportation of molecules, and defense mechanism. As new protein sequences are discovered on an everyday basis and protein databases continue to grow exponentially with time, analysis of protein families, understanding their evolutionary trends and detection of remote homologues have become extremely important. The traditional laboratory techniques of studying these proteins are very slow and time consuming. Therefore, biologists have turned to automated methods that are fast and capable of analyzing large amounts of data and determining relationships between proteins that would be difficult, if not impossible, for humans to identify through the traditional techniques.

Insertion/deletion (indel) and substitution of an amino acid are two common events that lead to the evolution of and variations in protein sequences. Further, many of the human diseases and functional divergence between homologous proteins are related more to the indel mutations than to the substitution mutations, even though the former occurs less often than the latter. A reliable detection of indels and their flanking regions is a major challenge in research related to protein evolution, structures and functions.

The first and most important step in studying a newly discovered protein sequence is to search protein databases for proteins that are similar or closely–related to the new protein, and then to align the new protein sequence to these proteins. Thus, the alignment of multiple protein sequences is one of the most commonly performed tasks in bioinformatics analyses, and has been used in many applications, including sequence annotation, phylogenetic tree estimation, evolutionary analysis, secondary structure prediction and protein database search. In spite of considerable research and efforts that have been recently deployed for improving the performance of multiple sequence alignment (MSA) algorithms, finding a highly accurate alignment between multiple protein sequences still remains a challenging problem.

The objectives of this thesis are to develop a novel scheme to predict indel flanking regions (IndelFRs) in a protein sequence and to develop an efficient algorithm for the alignment of multiple protein sequences incorporating the information on the predicted IndelFRs.

In the first part of the thesis, a variable–order Markov model–based scheme to predict indel flanking regions in a protein sequence for a given protein fold is proposed. In this scheme, two predictors, referred to as the PPM IndelFR and PST IndelFR predictors, are designed based on *prediction by partial match* and *probabilistic suffix tree*, respectively. The performance of the proposed IndelFR predictors is evaluated in terms of the commonly used metrics, namely, accuracy of prediction and F_1–measure. It is shown through extensive performance evaluation that the proposed predictors are able to predict IndelFRs in the protein sequences with high values of *accuracy* and F_1*–measure*. It is also shown that if one is interested only in predicting IndelFRs in protein sequences, it would be preferable to use the proposed predictors instead of HMMER 3.0 in view of the substantially superior performance of the former.

In the second part of the thesis, a novel and efficient algorithm incorporating

the information on the predicted IndelFRs for the alignment of multiple protein sequences is proposed. A new *variable gap penalty* function is introduced, which makes the gap placement in protein sequences more accurate for the protein alignment. The performance of the proposed alignment algorithm, named as MSAIndelFR algorithm, is evaluated in terms of the so called metrics, *sum-of-pairs* (SP) and *total columns* (TC). It is shown through extensive performance evaluation using four popular benchmarks, BAliBASE 3.0, OXBENCH, PREFAB 4.0, and SABmark 1.65, that the performance of MSAIndelFR is superior to that of the six most–widely used alignment algorithms, namely, Clustal W2, Clustal Omega, MSAProbs, Kalign2, MAFFT and MUSCLE.

Through the study undertaken in this thesis it is shown that a reliable detection of indels and their flanking regions can be achieved by using the proposed IndelFR predictors, and a substantial improvement in the protein alignment accuracy can be achieved by using the proposed variable gap penalty function. Thus, it is anticipated that this investigation will not only facilitate future studies on the modeling of indel mutations and protein sequence alignment, but will also open up new avenues for research concerning protein evolution, structures, and functions as well as for research concerning protein sequence alignment.

Acknowledgments

First, I thank God, the Almighty, for giving me the ability to finish my doctoral thesis. By God's will the completion of this work has become a reality. I would like to express my deep gratitude and appreciation to my supervisors, Professor M. Omair Ahmad and Professor M.N.S. Swamy, for their constant support, encouragement, patience, and invaluable guidance during the span of this research. I am grateful to them for providing me the freedom and motivation to explore new areas of research and new ideas. I also want to thank them for spending so many hours with me in correcting and improving the writing of this thesis. The useful suggestions provided by the committee members are also deeply appreciated.

I am indebted to Concordia University and NSERC for their financial supports that was so crucial for completing this research. I would also like to acknowledge the support of ReSMiQ. I would like to acknowledge the Department of Electrical and Computer Engineering at the Concordia University for its support and help in many ways, especially that of Kimberly R. Adams.

My sincere thanks go to my parents and family members for their support, encouragement, and constant prayers during my doctoral study. Special thanks are due to my dear wife, Asmaa, for her constant love, care, patience, encouragement, sacrifices and prayers during my doctoral study. She has provided me the peace of mind and the determination so crucial to complete my doctoral study. I would like to mention my beloved children, Oday and Qusay, whose beautiful smiling faces have always provided me the necessary strength and stamina to cape with my heavy workload.

Finally, I would also like to thank all my friends and colleagues who supported me and were always there in times of need.

Contents

List of Figures

List of Tables

LIST OF SYMBOLS

D	Memory length parameter for VOMM
g_e	Gap extension penalty
$g(k)$	Gap penalty for a gap of length k
g_o	Gap opening penalty
GPE_i	Gap extension penalty at position i in a protein sequence
GPO_i	Gap opening penalty at position i in a protein sequence
GPE_i^A	Gap extension penalty at position i in protein sequence A
GPO_i^A	Gap opening penalty at position i in protein sequence A
L	Length of a moving window
$loglossP(\mathbf{S}^n)$	Average log–loss function of the probability of the sequence $\mathbf{S}^n$
$LPPM_i$	Left PPM average log–loss value at position i
N_s	Number of occurrences of the context $\mathbf{s}$ in a training set
$N_{\mathbf{s}\sigma}$	Number of occurrences of the pattern $\mathbf{s}\sigma$ in a training set
N_T	Context threshold parameter
P	Precision (selectivity)
$P(\mathbf{S}^n)$	Probability of the sequence $\mathbf{S}^n$
$\widetilde{P}_k(\sigma\|\mathbf{s})$	Conditional empirical probability of observing σ given a context $\mathbf{s}$ of length k
$P_k(escape\|\mathbf{s})$	Probability mass assigned to all the amino acids that do not appear after the context $\mathbf{s} = s_{i-k}\cdots s_{i-1}$ of length k in a training set
$P(\mathbf{win}_i)$	Probability of a window of length L at position i
$P_k(\sigma\|\mathbf{s})$	Conditional probability of observing σ at position i, given a context $\mathbf{s} = s_{i-k}\cdots s_{i-1}$ of length k
R	Recall (sensitivity)

$RPPM_i$	Right PPM average log–loss value at position i
$\mathbf{S}^n$	Discrete sequence of length n
s_i	An element of $\mathbf{S}^n$
$\mathbf{set}_{ptr}$	Set of contexts
$suf(\mathbf{s})$	Parent or Suffix context of $\mathbf{s} = s_{i-k} \cdots s_{i-1}$
$\mathbf{win}_i$	Window of length L at position i
ϵ	Empty context
σ	A particular character/amino acid
Ψ	Finite alphabet set
$\Psi_{protein}$	Alphabet set containing all the twenty amino acid symbols
$\Psi_{\mathbf{s}}$	Set of amino acids appearing after the context $\mathbf{s} = s_{i-k} \cdots s_{i-1}$ of length k

LIST OF ACRONYMS

3D	Three dimensional
AGP	Affine gap penalty
DP	Dynamic programming
FFT	Fast Fourier transform
FN	False negative
FP	False positive
GPE	Gap extension penalty
GPO	Gap opening penalty
HMM	Hidden Markov model
indel	Insertion or deletion mutation
IndelFR	Indel flanking region
LFR	Left flanking region
LPPM	Left PPM
LPST	Left PST
MSA	Multiple sequence alignment
NW	Needleman and Wunsch algorithm
OF	Objective function
pHMM	Profile hidden Markov model
PPM	Prediction by partial match
PST	Probabilistic suffix tree
RFR	Right flanking region
RPPM	Right PPM
RPST	Right PST
SCOP	Structural classification of proteins database
SP	Sum–of–pairs

SW	Smith and Waterman algorithm
TC	Total columns
TN	True negative
TP	True positive
UPGMA	Unweighted pair group method with arithmetic mean
VGP	Variable gap penalty
VOMM	Variable–order Markov model

Chapter 1

Introduction

1.1 General

Proteins are the most important molecules inside every living organism, and they are actively involved in almost all of the cell functions, such as signal transmission, metabolic regulation, transportation of molecules, and defense mechanism. As the number of protein sequences in the protein databases is growing exponentially with time, analysis of protein families, understanding their evolutionary trends and detection of remote homologues[1] have become extremely important.A protein molecule is created in a cell as a chain of amino acids, called the polypeptide chain. A polypeptide chain can be represented as a string of characters by using the letter code of each amino acid. This string of characters is called the *primary structure* of a protein [1].

It is known that insertion/deletion (*indel*) and substitution of an amino acid are two common mutational events that lead to the evolution of and variations in protein sequences. It has been found that new proteins have evolved mainly through indel mutations [2, 3]. It should be noted that the insertion or deletion of an entire subsequence of amino acids often occurs as a single mutational event,

[1]The proteins that evolve from the same ancestor protein are called *homologous proteins*

and such single mutational event often affects several consecutive amino acids in a protein sequence [1]. Further, it has been found that differences among species, as well as many of the human diseases, are related more to the indel mutations than to the substitution mutations, even though the former occurs less often than the latter [4–6]. Therefore, developing methods that can efficiently detect indels and their flanking regions is extremely important in research related to protein evolution, structures and functions.

The alignment of multiple protein sequences is one of the most commonly performed tasks in bioinformatics analyses, and has been used in many applications, including sequence annotation, phylogenetic tree estimation, evolutionary analysis, secondary structure prediction and protein database search [7, 8]. In recent years, considerable effort has been devoted to the development of protein sequence alignment algorithms that can efficiently detect mutations and generate highly accurate alignment. Some of the most–widely used algorithms are Clustal W2 [9], Clustal Omega [10], Kalign2 [11], MSAProbs [12], MAFFT [13,14] and MUSCLE [15]. From the biological point of view, the alignment of protein sequences is motivated by the following facts: (i) all living organisms are related by evolution, (ii) every protein sequence has an evolutionary history, and (iii) every protein sequence contains biologically important regions that are less likely to mutate than others, and thus, finding protein conserved regions might be a strong indication to a functionally important region. Therefore, the proteins that are closely related should have higher similarity than those which are not closely related. In the other words, the sequence similarity implies functional similarity which, in turn, is likely to indicate a common ancestor. As a consequence, biologists can transfer all information (structure and function) known about a given protein to the newly discovered protein, if a high similarity between them is found.

Multiple sequence alignment (MSA) allows us to identify parts of the protein

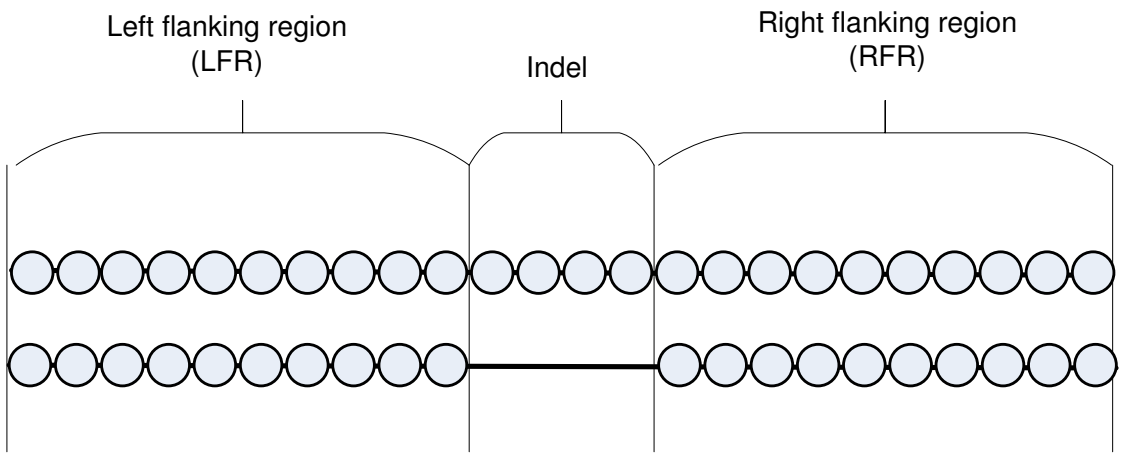


Figure 1.1: Indel and the flanking regions.

sequences that are similar to one another with gaps (spaces) inserted in such a way that similar parts of these sequences can be easily identified [16]. The concept of a gap in an alignment is important, since the gap locations indicate the locations of indel mutations in protein sequences. It should be noted that the insertion or deletion of an entire subsequence often occurs as a single mutational event, and such single mutational events can create gaps of varying sizes [1]. When a pair of protein sequences is aligned, a gap in any of the two sequences is defined as an *indel region*. Segments of protein sequence immediately before and after an indel region are called *flanking regions* as shown in Figure 1.1. It has been found that there exists a strong relationship between indels and their flanking regions [3, 5, 17–19]. In this thesis, an indel along with its left and right flanking regions is referred to as an *indel flanking region (IndelFR)*.

Normally, an alignment between protein sequences is intended to reflect the cost of mutational events needed to transform one sequence to another [7, 20]. Therefore, MSA algorithms use an *objective function* (OF) to measure the quality of an alignment by assigning a quality score to every possible alignment of the input sequences. Defining a proper OF is a non–trivial task and has been a subject of active research.

In theory, an OF should incorporates information about the sequences, such as their secondary structures, solvent accessibility, hydrophobic indices, function and evolutionary history. Information regarding the above is not always available and difficult to use, even if available. Hence, the MSA algorithms use a simple OF consisting of a *gap penalty function* to score the gaps and substitution matrices to measure the similarity of amino acid pairs. The score of a given alignment is then defined as the sum of the similarity scores of the aligned amino acids minus the corresponding penalty for every gap.

The most popular substitution matrices are BLOSUM [21], GONNET [22] and PAM [23]. The substitution matrices assign a positive value if similar amino acids are aligned, and a negative value if dissimilar amino acids are aligned. It has been shown in [24] that the selection of a particular substitution matrix does not noticeably affect the alignment accuracy, and that there is little difference in the alignment accuracy using BLOSUM, PAM or GONNET as the substitution matrix. Therefore, there has been a growing interest to propose an appropriate gap penalty function that can improve both the objective function and the alignment accuracy.

The most widely used gap penalty function is the *affine gap penalty* (AGP), given by $g(k) = g_o + kg_e$ for a gap of length k. The function $g(k)$ involves two parameters g_o and g_e, representing a gap opening penalty at a specific position in the protein sequence and representing an extension penalty for extending the gap, respectively. This linear AGP function has the advantage of simplicity and ease of use in MSA algorithms. However, this penalty function is restrictive in the sense that the two parameters remain fixed for aligning different positions in the protein sequence. There are a few strategies that have been proposed for improving the gap penalty functions. In the past two decades, several groups have attempted to find the distribution of indel lengths for a more effective gap penalty function [25] or to empirically estimate the parameters for AGP [26,27]. Although there is no consensus

as to the distribution of indel lengths to determine the optimal form of gap penalty, a few gap penalty functions have been proposed for improving the alignment accuracy, including the generalized AGP [28, 29], a long indel model [30], and a logarithmic gap penalty [31]. It should be noted that these gap penalty functions have not been used widely, since they are difficult to implement and provide only a limited improvement to the alignment accuracy.

MSAProbs and Kalign2 are MSA algorithms for which an AGP function is used. In MSAprobs, fixed parameters are used for the AGP function, wherein a gap opening penalty of 22 and a gap extension penalty of 1 are used by default [12]. In Kalign2, an AGP function is used to align sequences/profiles, and the users are allowed to specify two additional parameters, a *terminal gap penalty* that is used to penalize N/C–terminal gaps in protein sequences, and a *gap_inc* that is used to increase the gap opening and extension penalties depending on the number of existing gaps in a given profile. It should be noted that Kalign2 determines the default gap penalties for protein alignments by training on a BAliBASE 3.0 benchmark [32] in order to obtain optimal alignment results. In the MAFFT, MUSCLE, Clustal W2 and Clustal Omega MSA algorithms, a gap opening penalty (GPO) and a gap extension penalty (GPE) values are initially specified; then, these algorithms automatically attempt to choose appropriate gap penalties according to some specific rules. The algorithms MAFFT and MUSCLE use an AGP function, wherein the default values are modified depending on the number of existing gaps at a particular position for a given profile [14, 15]. Clustal W2 and Clustal Omega use an AGP function, wherein a gap opening penalty (GPO) and a gap extension penalty (GPE) are initially set by the user from a menu, and then, these algorithms automatically attempt to choose appropriate gap penalties for each sequence alignment according to the features of the input sequences, such as sequence divergence, length, and local hydrophobic amino acids. It should be noted that the choice of the AGP parame-

ters has a substantial effect on the alignment accuracy [33–35], and the widely–used AGP works well for closely related or similar sequences, but they are less effective for highly diverged or dissimilar sequences. As a consequence, there has been a growing interest in conducting multiple sequence alignment with more general and flexible gap penalty functions.

1.2 Motivation and Objectives of the Thesis

As discussed in the previous section, developing methods that can efficiently predict IndelFRs is a major challenge in research related to protein evolution, structures and functions, especially with the rapid growth in the number of protein sequences in the protein databases. In the protein alignment, the gap regions are introduced to indicate the locations of IndelFRs in protein sequences. It is known that the accuracy of the protein alignment algorithm in introducing gap regions is extremely sensitive to the choice of the gap penalty function. There are a few strategies that have been proposed to improve the gap penalty functions (See section 1.1); however, none of these take advantage of the strong relationship that exists between indels and their flanking regions as has been shown in [3, 5, 17–19]. Therefore, it is extremely important to develop new methods that can predict IndelFRs in the protein sequences without relying on the protein sequence alignment and to propose new gap penalty functions taking advantage of the relationship between indels and their flanking regions.

The objectives of this thesis are two–fold: (i) to develop an efficient and reliable method to predict IndelFRs that takes advantage of the strong relationship that exist between indels and their flanking regions, and (ii) to develop an efficient algorithm for the alignment of multiple protein sequences incorporating the information on the predicted IndelFRs. In order to achieve these objectives, the thesis is divided into two parts.

In the first part of the thesis, a variable-order Markov model–based scheme to predict indel flanking regions in a protein sequence for a given protein fold is proposed [36]. In this scheme, two predictors, referred to as the PPM IndelFR and PST IndelFR predictors, based on *prediction by partial match* [37] and *probabilistic suffix tree* [38], respectively, are designed. The proposed IndelFR predictors are trained using the flanking regions that are listed in the IndelFR database [17]. In order to demonstrate the effectiveness of the proposed IndelFR predictors in predicting IndelFRs for a given protein fold, extensive experiments are carried out using the IndelFR database and the sequence alignment benchmark (SABmark 1.65) [39]. Furthermore, the performance of the two proposed predictors is compared with that using the latest version of the alignment software HMMER, HMMER 3.0 [40]. The HMMER algorithms are considered to be the most accurate algorithms in term of detecting indel mutations in protein sequences. It is shown through extensive performance evaluation that the proposed predictors are able to predict IndelFRs in the protein sequences with high values of *accuracy* and F_1*–measure*. It is shown that if one is interested only in predicting IndelFRs in protein sequences, it would be preferable to use the proposed predictors instead of HMMER 3.0 in view of the substantially superior performance of the former.

In the second part of the thesis, a novel and efficient algorithm incorporating the information on the predicted IndelFRs for the alignment of multiple protein sequences is proposed. The key innovation in the proposed algorithm, referred to as MSAIndelFR algorithm, is the use of the predicted information about IndelFRs to propose a new *variable gap penalty* (VGP) function, wherein the gap opening penalty is position–specific and the gap extension penalty is region–specific. It should be noted that the predicted IndelFRs are the most likely regions for the gaps to be introduced in the protein sequence alignment, since they are strongly related to indel mutations [3, 5, 17–19]. Therefore, it is expected that more accurate alignments can

be achieved by integrating the predicted information about IndelFRs into the gap penalty function. Extensive experiments are conducted on four popular alignment benchmarks, namely, BAliBASE 3.0 [32], OXBENCH [41], PREFAB 4.0 [15], and SABmark 1.65 [39], to show the effectiveness of the proposed MSAIndelFR algorithm in generating protein alignment. It is shown that the performance of MSAIndelFR is superior to that of the six most–widely used alignment algorithms, namely, Clustal W2 [9], Clustal Omega [10], Kalign2 [11], MSAProbs [12], MAFFT [13, 14] and MUSCLE [15].

1.3 Thesis Organization

The thesis is organized as follows.

In Chapter 2, the background material necessary for the research work undertaken in this thesis is given. The chapter begins with an introduction to protein sequences. This is followed by a brief introduction to variable–order Markov model (VOMM) and profile hidden Markov model (pHMM), and the related notations and concepts. Finally, a brief review for some of the most-widely used multiple sequence alignment (MSA) algorithms and popular alignment benchmarks is given. This review is intended to facilitate the understanding of the development of the algorithm presented in the thesis.

In Chapter 3, a variable-order Markov model–based scheme to predict indel flanking regions in a protein sequence for a given protein fold is proposed [36]. In this scheme, two predictors, referred to as the PPM IndelFR and PST IndelFR predictors, are designed based on *prediction by partial match* [37] and *probabilistic suffix tree* [38], respectively. Simulation results showing that the best choice for the memory length D is 4 for all the selected protein folds, and that the proposed predictors are able to predict IndelFRs in the protein sequences with high values of accuracy and F_1–measure are presented. It is also shown that the proposed IndelFR

predictor predicts the indel flanking regions more accurately than the HMMER 3.0 does.

In Chapter 4, a novel and efficient algorithm that incorporates the information on the predicted IndelFRs for the alignment of multiple protein sequences is proposed. The performance of the proposed algorithm is studied using four popular alignment benchmarks, namely, BAliBASE 3.0 [32], OXBENCH [41], PREFAB 4.0 [15], and SABmark 1.65 [39]. Its performance is also compared with the six most–widely used alignment algorithms in terms of the *sum–of–pairs* and *total column* metrics.

Finally, some concluding remarks highlighting the contributions of the thesis and some suggestions for future work based on the schemes developed in this thesis are provided in Chapter 5.

Chapter 2

Background Material

In this chapter, some background material necessary for the understanding and development of the work undertaken in this thesis is presented. A brief introduction to protein sequences, notations and concepts that are related to variable–order Markov model (VOMM) and profile hidden Markov model (pHMM) are given. Some of the important sequence alignment algorithms and the four most popular alignment benchmarks are briefly reviewed.

2.1 Proteins

Proteins are the most important molecules in living organism, and they are involved in every function of the cells, such as signal transmission, metabolic regulation, transportation of molecules, and defense mechanisms. A protein molecule is a chain of *amino acids*, also known as the polypeptide chain. There are 20 different amino acids that are involved in building any protein molecule. The names of these amino acids and their one–letter codes are listed in Table 2.1. A polypeptide chain can be represented as a string of characters by writing down the one–letter code for each amino acid. This string represents the *primary structure* of the protein [1]. In

Table 2.1
Names and one–letter codes of twenty different amino acids

Amino acid name	**code**	**Amino acid name**	**code**
Alanine	A	Tryptophan	W
Valine	V	Tyrosine	Y
Leucine	L	Asparagine	N
Isoleucine	I	Glutamine	Q
Phenylalanine	F	Aspartic acid	D
Proline	P	Glutamic acid	E
Methionine	M	Lysine	K
Serine	S	Arginine	R
Threonine	T	Histidine	H
Cysteine	C	Glycine	G

addition to the primary structure, a protein has secondary and tertiary structures [1]. *Secondary structure* of a protein refers to well–determined local sequence elements, such as an *alpha helix*, a *beta strand* or any other local sequence elements that are neither a helix nor a strand. These other local sequences, usually called *loops* or *coils*, may have a large variety of shapes. These secondary structure elements of a protein can be combined together to create a motif, which is a simple combination of a few consecutive secondary structure elements with a specific geometric arrangement, such as *helix-loop-helix* or *strand-loop-helix*. Some, but not all, motifs are associated with specific biological functions. The *tertiary structure* of a protein refers to the three–dimensional structure of the protein, where the secondary structure elements form the physical core of the three–dimensional structure, and loops are located on the surface of the tertiary structure. A *domain* refers to a combination of several secondary elements and motifs, which may not necessarily be contiguous and which are usually packed in a compact structure. A protein may contain a single domain or several different domains, or several copies of the same domain. It should be noted that the proteins that evolve from the same ancestor protein are called *homologous proteins*, and they usually have similar structural and functional properties [1, 7, 8].

Normally, the proteins are classified into families based on the existence of a specific motif or domain in their structure, where the existence of such a motif or domain has a major indication about the biological role of the protein. The *structural classification of proteins* (SCOP) database contains a comprehensive classification of all the proteins of the known structures, according to their evolutionary and structural relationships [42]. In this database, the proteins have been classified into families, superfamilies, common fold and, finally, into classes at the top level of the structural hierarchy.

As new protein sequences are discovered on an everyday basis and protein data–bases continue to grow exponentially with time, analysis of protein families, understanding their evolutionary trends and detection of remote homologues have become extremely important. It is known that insertion/deletion (*indel*) and substitution of an amino acid are two common mutational events that lead to the evolution of and variations in protein sequences. It has been found that new proteins have evolved mainly through indel mutations [2,3]. It should be noted that the insertion or deletion of an entire subsequence of amino acids often occurs as a single mutational event, and such single mutational event often affects several adjacent amino acids in a protein sequence [1]. Indel mutations have been found to occur more often in the loop regions [43,44], and mainly in essential proteins and in those proteins that interact highly with others [45]. The functional divergence between homologous proteins may also be caused by indel mutations that occur in the regions between secondary structures of a protein [46]. Further, it has been found that differences among species, as well as many of the human diseases, are related to indel mutations, which occur less often than substitution mutations do [4–6]. Therefore, developing methods that can efficiently predict indel mutations is extremely important in research related to protein evolution, structures and functions.

2.2 Variable-Order Markov Model (VOMM)

In this section, we introduce some preliminary notations and definitions that are used to build the theory of variable–order Markov model (VOMM). Let Ψ be a finite alphabet set, and $\mathbf{S}^n = s_1 s_2 s_3 \cdots s_n$ be a discrete sequence of length n, where s_i, $1 \leq i \leq n$, can be one of the characters that exist in the finite alphabet set Ψ (i.e., $s_i \in \Psi$). VOMM provides the conditional probabilities of observing a particular character at a certain position in the sequence given contexts of varying lengths. A context represents all the previously observed characters before the particular character in question is observed at that position. The context length varies from zero to the memory length of the VOMM.

In VOMM, the conditional probability $P_k(\sigma|s_{i-k}\cdots s_{i-1})$ of observing a particular character σ at position i, given a context $\mathbf{s} = s_{i-k}\cdots s_{i-1}$ of length k, where the context $\mathbf{s} = s_{i-k}\cdots s_{i-1}$ represents all the previously observed characters before σ is observed at position i, each $s_j, j \in i-1, \cdots, i-k$, representing one of the possible characters that exist in the alphabet set Ψ. The context length k varies from zero to D, where D is the memory length of the VOMM. For VOMM, $P(\mathbf{S}^n)$ is used to denote the probability of the sequence $\mathbf{S}^n = s_1 s_2 s_3 \cdots s_n$. If VOMM has a memory length D, $P(\mathbf{S}^n)$ is given by

$$P(\mathbf{S}^n) = P_0(s_1)P_1(s_2|s_1)\cdots P_D(s_{D+1}|s_1 s_2 \cdots s_D)\cdots P_D(s_n|s_{n-D}\cdots s_{n-1}) \qquad (2.1)$$

Maximizing the probability $P(\mathbf{S}^n)$ is equivalent to minimizing the average log–loss function defined as [1]

$$
\begin{aligned}
loglossP(\mathbf{S}^n) = -\frac{1}{n}\Bigg(& \log P_0(s_1) + \log P_1(s_2|s_1) + \cdots + \\
& \log P_D(s_{D+1}|s_1 s_2 \cdots s_D) + \cdots + \\
& \log P_D(s_n|s_{n-D} \cdots s_{n-1}) \Bigg)
\end{aligned} \tag{2.2}
$$

where the logarithm taken with a base of 2.

Over the years, many VOMM structures, such as Lampel-Ziv compression [47], context tree weighting [48], prediction by partial match (PPM) [37] and probabilistic suffix tree (PST) [38,49], have been proposed to efficiently minimize the average log–loss function given by (2.2) . In the next chapter, we will discuss only two of them, namely, PPM and PST, to propose two predictors that can predict indel flanking regions in a protein sequence. It should be noted that PPM and PST are among the most commonly–employed structures in prediction applications. The effectiveness of PPM and PST for prediction of sequences in various applications has been examined in [50]. PST has also been used to model DNA sequences in [38]. Further, it has been used in modeling and prediction of protein families in [51].

2.3 Profile Hidden Markov Model (pHMM)

Normally, proteins are classified into families based on the existence of a specific motif or domain in their structure, where the existence of a specific motif or domain has a major indication about the biological role of the protein. Once all the protein members of a given protein family are aligned, it is seen that some regions are more conserved than others, whereas some are more prone to indel mutations than others. This variation among the protein members of a given protein family can be described by using the profile hidden Markov model (pHMM) [20, 52, 53]. The pHMM encodes position–specific information about the frequency of a particular

amino acid as well as the frequency of an insertion or deletion at a specific position for a given protein family. The goal of the pHMM is to model each protein family in such a way that unique patterns are presented with high probability, whereas variation is allowed with low probability. By using position–variant probability to score indel mutations, the pHMM is able to utilize the fact that indel mutations occur more frequently in some parts of a protein sequence than in others (e.g., in loop regions) [43, 44]. Therefore, the pHMM–based alignment algorithms are considered to be the most accurate algorithms in detecting indel mutations in protein sequences.

Several available software packages, such as HMMER [40] and SAM [54, 55] have implemented pHMM–based alignment algorithms. Among these packages, HMMER is the most–commonly used software package in protein database search and comparison. HMMER produces a pHMM that describes the probabilities of occurrence of different amino acids at a given position of the aligned sequences. A new protein sequence can then be aligned to this profile model rather than aligning it directly with the sequences used to produce the pHMM. A sequence–to–pHMM alignment is more accurate than a sequence–to–sequence alignment, since the pHMM is built from a high–quality multiple sequence alignment of protein sequences belonging to a family. Moreover, the model estimates the likelihood that the new sequence is a member in the protein family used to produce the model. A collection of pHMMs covering many protein families have been generated using HMMER and they are available in the Pfam database [56]. The disadvantage of using pHMM–based alignment algorithms for detecting mutations is that they assume the occurrence of mutations in the protein sequence to follow the first-order Markov chain. Example 1 given below illustrates how pHMM can be constructed from MSA of protein sequences.

Example 1.

A pHMM can be constructed from existing multiple alignments of homologous protein sequences. The pHMM contains three kinds of states represented by three different shapes: squares, diamonds, and circles. Squares represent *match states*, and amino acids emitted from the match states form the conserved characteristics of aligned protein sequences, such as motifs. The diamond shapes represent *insert states*, and amino acids emitted from the insert states model insertion mutations. Circles represent *delete states (silent states)*, and these states model deletion mutations. It should be noted that transitions from state to state progress from left to right through the model, with the exception of the self-loops on the insertion states. Hence, certain paths allow us to insert amino acids in the alignment.

Given the protein sequence alignment shown in Figure 2.1, it is seen that these three sequences have many features in common. They all start with the same four amino acids, VIVA, and they have a position where various choices are possible, then they have another conserved position, A. After a variable number of positions, more or less rigidly specified, they all have the symbol S at the end. The pHMM can be constructed as shown in Figure 2.2.

We write only the dominant amino acid symbols in the match state. It should be clear that, in general, any amino acid is possible to be emitted with different probabilities. Transitions with low probability are denoted by dotted lines and those with high probability by solid lines. The beginning and end states are added to pHMM to specify the boundaries of the pHMM, and they are indicated by **B** and **E**, respectively. Every path between the beginning and end states represents a possible alignment of any protein sequence with pHMM.

■

```
V I V A L A S V E G A S
V I V A D A - V I - - S
V I V A D A L L - - A S
```

Figure 2.1: A part of protein multiple sequence alignment containing twelve positions.

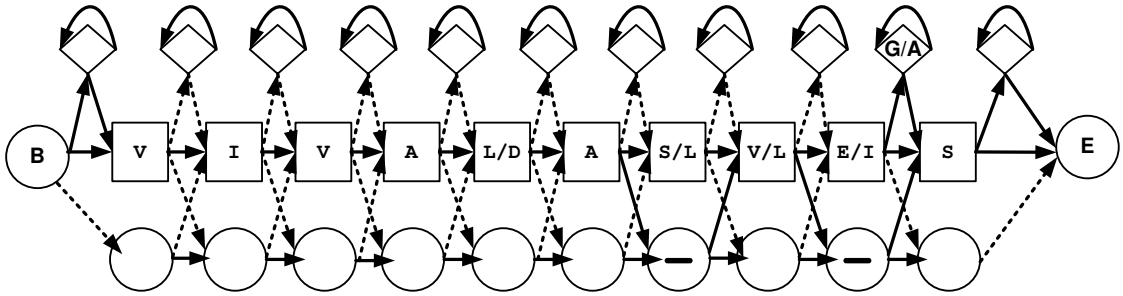


Figure 2.2: A pHMM corresponding to the protein multiple alignments shown in Figure 2.1.

2.4 Multiple Sequence Alignment Algorithms

For a given set of protein sequences, the overall goal of multiple sequence alignment (MSA) is to identify parts of these sequences that are similar to one another. It should be noted that the similarity at the protein primary structure may indicate that these sequences originated from a common ancestor by indel and (or) substitution mutations [7,8]. Therefore, MSA is a crucial step in bioinformatic analyses, and is used in many applications including sequence annotation, phylogenetic tree estimation, evolutionary analysis, secondary structure prediction and protein database search [7, 33, 57]. MSA allows us to identify parts of the protein sequences that are similar to one another with gaps (spaces) inserted in such a way that similar parts of these sequences can be easily identified [16]. The concept of a gap in an alignment is important, since the gap locations indicate the locations of indel mutation events

in protein sequences. It should be noted that the insertion or deletion of an entire subsequence often occurs as a single mutational event, and such single mutational events can create gaps of varying sizes [1].

To find the globally optimum alignment for a given set of protein sequences, a dynamic programming technique can be used. There are two major challenges related to finding an optimum MSA. First, finding an optimum alignment for n sequences using the dynamic programming approach is an NP–complete problem. Therefore, many MSA algorithms are based on a heuristic search for the optimum MSA. Second, a perfect *objective function* (OF) does not yet exist. In theory, an OF should incorporate information about the sequences, such as their secondary structures, solvent accessibility, hydrophobic indices, function and evolutionary history. Information regarding the above is not always available and difficult to use even if available.

In order to meet the first challenge, considerable effort has been devoted to the development of MSA algorithms that can efficiently detect mutations and generate highly accurate alignments. Some of the significant algorithms are Clustal W2 [9], Clustal Omega [10], MSAProbs [12], Kalign2 [11], MAFFT [13,14,58] and MUSCLE [59].

In order to meet the second challenge, many of the MSA algorithms use an objective function consisting of a *gap penalty function* to score the gaps and *substitution matrices* to measure the similarity of amino acid pairs. The most popular substitution matrices are BLOSUM [21], PAM [23] and GONNET [22]. These matrices present a measurement of how probable it is that a certain amino acid will mutate to other amino acids. The substitution matrices assign a positive value if similar amino acids are aligned, and a negative value if dissimilar amino acids are aligned. It has been shown in [24] that the selection of a particular substitution matrix does not noticeably affect the alignment accuracy, and that there is little difference in

the alignment accuracy using BLOSUM, PAM or GONNET as the substitution matrix. Therefore, a considerable amount of effort has been directed to propose an appropriate gap penalty function that can improve both OF and the alignment accuracy.

The most widely used gap penalty function is the *affine gap penalty* (AGP), given by $g(k) = g_o + kg_e$ for a gap of length k. The function $g(k)$ involves two parameters g_o and g_e, representing, respectively, a gap opening penalty at a specific position in the protein sequence and an extension penalty for extending the gap. This linear AGP function has the advantage of simplicity and ease of use in MSA algorithms. However, this penalty function is restrictive in the sense that the two parameters remain fixed for aligning different positions in the protein sequence. There are a few strategies that have been proposed for improving the gap penalty functions. In the past two decades, several groups have attempted to find the distribution of indel lengths for a more effective gap penalty function [25] or to empirically estimate the parameters for AGP [26, 27]. Although there is no consensus as to the distribution of indel lengths to determine the optimal form of gap penalty, a few gap penalty functions have been proposed for improving the alignment accuracy, including the generalized AGP [28,29], a long indel model [30], and a logarithmic gap penalty [31]. However, these penalty functions have not been used widely by others, since they are difficult to implement and they provide only limited improvement in alignment accuracy.

Now, we will review the significant MSA algorithms that we have mentioned earlier. Clustal W2, Clustal Omega, Kalign2, and MSAProbs are *progressive alignment* algorithms, while MAFFT and MUSCLE generate an initial alignment using the progressive alignment algorithm and then, iteratively refine this alignment to achieve higher alignment accuracy. A progressive alignment algorithm involves three steps: (i) calculation of the pairwise distances between all pairs of sequences to de-

termine the similarity of each pair of sequences, (ii) construction of a guide tree based on the distance matrix, and finally, (iii) alignment of the sequences according to an order determined by the guide tree [1, 60]. In the final step, the algorithm follows the guide tree to add the sequences one by one into MSA from the leaves (sequences) toward the root. Hence, each node in the tree will contain an alignment resulting from the pairwise alignment of its two children. The pairwise sequence alignment algorithm used in the progressive alignment algorithm must be able to align two sequences, a sequence to an alignment (profile), and pairs of profiles.

Clustal W2 and Clustal Omega are derived from Clustal W [61]. Clustal W is historically one of the most popular MSA algorithms, wherein the sequences or profiles are added to the alignment according to the order indicated by the pre-computed guide tree. Clustal W calculates the pairwise distances between all the pairs of sequences using the k–tuple method [62], and then constructs the guide tree using the neighbor–joining method [63]. In Clustal W, a memory–efficient dynamic programming algorithm proposed in [64] is used to align the sequences or profiles. Clustal W2 uses the *unweighted pair group method with arithmetic mean* (UPGMA) [65] to construct the guide tree. This helps in increasing the speed of the alignment over that of Clustal W for a large number of protein sequences. Clustal Omega is the latest MSA algorithm in the Clustal family, and the main improvements of Clustal Omega over Clustal W2 are as follows: (i) it can align any number of protein sequences, and (ii) it allows the use of a profile hidden Markov model, derived from an alignment of protein sequences related to the input sequences. Further, Clustal Omega is the most accurate and scalable MSA algorithm amongst the Clustal family.

Kalign2 [11] is an enhanced version of Kalign [66], which is a progressive alignment algorithm, wherein the pairwise distances between all pairs of sequences are estimated based on the Wu–Manber approximate string matching algorithm [67] and the guide tree constructed using UPGMA. Kalign uses the dynamic programming

algorithm proposed in [64] to align sequences/profiles. In Kalign 2, the Wu–Manber algorithm is replaced by the faster Muth–Manber string matching algorithm [68]. This helps in increasing the speed of calculating the pairwise distances between all the pairs of sequences. MSAProbs [12] is based on combining a pair hidden Markov model with partition functions to calculate the posterior probabilities, which are used in estimating the pairwise distance matrix. In MSAProbs, the guide tree constructed using UPGMA and the sequences or profiles are aligned using the standard dynamic programming approach.

The alignment algorithms MAFFT and MUSCLE are not fully progressive. MAFFT uses the fast Fourier transform for a rapid identification of similar regions in the protein sequences. It then iteratively refines the alignment results after performing an initial progressive alignment. In MAFFT, the protein sequences are converted to a sequence composed of volume and polarity values of each amino acid, and the guide tree constructed using UPGMA. MUSCLE works by iteratively refining the alignment results with progressive alignment at the core. MUSCLE constructs an initial guide tree and produces an initial alignment, then uses this alignment to construct a new guide tree. The newly constructed guide tree is finally used to produce a new alignment. This process is repeated for a fixed number of iterations or until the alignment converges. In MUSCLE, the guide trees are produced using UPGMA. It should be noted that MAFTT and MUSCLE use the standard dynamic programming approach to align the sequences or profiles.

MSAProbs and Kalign2 are MSA algorithms for which an AGP function is used. In MSAprobs, fixed parameters are used for the AGP function, wherein a gap opening penalty of 22 and a gap extension penalty of 1 are used by default [12]. In Kalign2, an AGP function is used to align sequences/profiles, and the users are allowed to specify two additional parameters, a *terminal gap penalty* and a *gap_inc*; the terminal gap penalty is used to penalize N/C–terminal gaps in protein sequences,

and the gap_inc is used to increase the gap opening and extension penalties depending on the number of existing gaps in a given profile. It should be noted that Kalign 2 determines the default gap penalties for protein alignments by training on a BAliBASE 3.0 benchmark [32] in order to obtain optimal alignment results. In the MAFFT, MUSCLE, Clustal W2 and Clustal Omega MSA algorithms, a gap opening penalty (GPO) and a gap extension penalty (GPE) values are initially specified; then, these algorithms automatically attempt to choose appropriate gap penalties according to some specific rules. The algorithms MAFFT and MUSCLE use an AGP function, wherein the default values are modified depending on the number of existing gaps at a particular position for a given profile [14, 15]. Clustal W2 and Clustal Omega use an AGP function, wherein a gap opening penalty (GPO) and a gap extension penalty (GPE) are initially set by the user from a menu, and then, these algorithms automatically attempt to choose appropriate gap penalties for each sequence alignment according to the features of the input sequences, such as sequence divergence, length, and local hydrophobic amino acids. It should be noted that the choice of the AGP parameters has a substantial effect on the alignment accuracy [33–35], and the widely–used AGP works well for closely related or similar sequences, but they are less effective for highly diverged or dissimilar sequences. As a consequence, there has been a growing interest in conducting multiple sequence alignment with more general and flexible gap penalty functions.

2.5 Alignment Benchmarks

The performance of MSA algorithms are usually evaluated and compared to each other based on alignment benchmarks containing reference alignments. The four most popular benchmarks are BAliBASE 3.0 [32], OXBENCH [41], PREFAB 4.0 [59] and SABmark 1.65 [39]. It has been found that the homologous proteins often retain similar 3D structures, even though their primary structures do not show

a significant similarity [7, 8]. Therefore, structure–based alignment is considered to be more correct from an evolutionary point of view than its sequence–based counterpart. In view of this, the above four benchmarks mainly contain structure–based alignments. It should be noted that these benchmarks define *core blocks* for each reference alignment, where core blocks in the reference alignment refer to regions for which reliable alignments are known to exist. These core blocks are used to evaluate and compare the performance of the various MSA algorithms.

2.5.1 BAliBASE 3.0

BAliBASE 3.0 Benchmark is the most widely used benchmark for evaluating multiple protein sequence alignment algorithms. Each reference alignment is constructed using 3D structural alignment algorithm SSAP [69], and then manually verified to ensure that the secondary structure elements are aligned correctly. It should be noted that for each reference alignment, core blocks are determined to exclude the regions for which the 3D structure is unreliable, for example, the borders of secondary structure elements or in loop regions.

BAliBASE 3.0 contains 218 reference alignments, where 168 reference alignments are provided in two versions, one where protein sequences are described as *truncated to homologous regions* and one where protein are described as *full-length sequences* (untrimmed). Hence, we have 386 reference alignments in BAliBASE 3.0. These reference alignments are organized into reference sets that are designed to represent real multiple alignment problems. Each reference set represents some characteristics such as long or short sequences, high or low sequence identity, large N/C terminal extensions or large internal insertions.

2.5.2 OXBENCH

OXBENCH benchmark is a set of structure–based alignments generated using multiple structure alignment algorithm STAMP [70], where the protein structures are taken from the 3Dee database [71]. 3Dee contains 3-dimensional structure of proteins that are experimentally determined. The OXBENCH benchmark uses 218 distinct protein families to generate 395 reference alignments, where the number of protein sequences in each alignment varies from 2 to 122 .

2.5.3 PREFAB 4.0

PREFAB 4.0 benchmark is a fully automatically generated benchmark containing 1681 reference alignments. Pairs of sequences with known 3D structures have been selected and aligned using two different 3D structure alignment algorithms, FSSP [72] and CE [73], and then the set of regions on which the two structural alignments agree are retained. Each sequence in the aligned pair is then used to query a database using PSI-BLAST [74], from which high–scoring hits are collected. Finally, the queries and their hits are combined to make test sets of 50 sequences.

2.5.4 SABmark 1.65

SABmark 1.65 is a very challenging benchmark for multiple sequence alignment according to a comprehensive study [75]. This benchmark is divided into two subsets: *Twilight zone* and *Superfamilies*. The similarity level between any two protein sequences is less than 50% in the Superfamily set, while it is at most 25% in the Twilight set. The pairwise reference alignments are constructed using two different 3D structure alignment algorithms, SOFI [76] and CE [73]. In [77], the author has argued that the pairwise reference alignments in SABmark 1.65 are not suitable to evaluate the MSA algorithms, and hence, has constructed the SABRE benchmark

(*http://www.drive5.com/bench*) containing 423 out of the 634 SABmark groups.

2.6 Summary

In this chapter, the background material necessary for the research work undertaken in this this has been presented. A brief introduction to protein sequences, and concepts that are related to variable–order Markov model (VOMM) and profile hidden Markov model (pHMM) have been reviewed. An example of constructing pHMM from multiple protein sequence alignment has been presented. The advantages and disadvantages of using pHMM in protein sequence alignment have also been discussed.

Some of the most-widely used MSA algorithms have been briefly reviewed, and their alignment strategies to generate MSA have been discussed. These algorithms are seen to follow essentially the progressive alignment scheme to find MSA, and they depend on the dynamic programming approach to generate pairwise alignment between sequences/profiles. It has been noted that these MSA algorithms have selected the affine gap penalty function to score gaps. Finally, the most popular alignment benchmarks that are used to measure the performance of MSA algorithms have been briefly discussed.

Chapter 3

Prediction of Indel Flanking Regions in Protein Sequences using a Variable-Order Markov Model

3.1 Introduction

In this chapter, we propose a variable-order Markov model–based scheme to predict indel flanking regions (IndelFRs) in a protein sequence for a given protein fold [36]. It is recalled that we refer to an indel along with its left and right flanking regions as an *indel flanking region*. In this proposed scheme, two predictors, referred to as the PPM IndelFR and PST IndelFR predictors, are designed based on *prediction by partial match* (PPM) [37] and *probabilistic suffix tree* (PST) [38], respectively.

This chapter starts by classifying the indel flanking regions according to the number of amino acids in their flanking regions in Section 3.2. In Section 3.3, a variable-order Markov model–based predictors are designed to detect the locations of IndelFRs in a protein sequence for a given protein fold. In Section 3.4, using the proposed predictors, an algorithm is then developed to identify the locations of IndelFRs in a protein sequence. In Section refsec:ch3results, experiments are carried

out to study the performance of the proposed predictors by employing them on IndelFR database and using the sequence alignment benchmark (SABmark 1.65) as reference for determining the locations of IndelFRs. Finally, Section 3.6 summarizes the work of this chapter along with some concluding remarks on the features of the proposed predictors.

3.2 Segments of Indel Flanking Regions

In the IndelFR database, indels and their flanking regions are extracted from alignments by dividing equally the region between two adjacent indels, and by taking 10 amino acids as the upper limit for the flanking regions. It has been shown in [3] that the impact of an indel on its flanking regions reduces dramatically as we move away from the indel, and this impact is negligible after 10 amino acids. In the present study, we classify the indel regions stored in the IndelFR database according to the number of amino acids in the flanking regions as follows.

i. If the number of amino acids between two indels is greater than 20, then we consider each of the two flanking regions between them to have exactly 10 amino acids.

ii. If the number of amino acids between two indels is less than or equal 20, but greater than or equal 2, we still consider these two indels as two separate indels, and the region between them split equally or as equally as possible to define the flanking regions between the two indels.

iii. If the number of amino acids between two adjacent indels in the same sequence is unity, then we combine the two indels along with the single amino acid in between to treat the combination as a single indel.

iv. If the number of amino acids between two adjacent indels that are not in the

same sequence is unity, then we treat these two indels as distinct. Thus, the right (left) flanking region of one of the indels and the left (right) flanking region of the other indel would each have only one amino acid.

We refer to an indel along with its left and right flanking regions as an *indel flanking region (IndelFR)*. Figure 3.1 shows an example illustrating each of the above situations. It is noted that there are 3 IndelFRs for each of the two alignments shown in Figure 3.1.

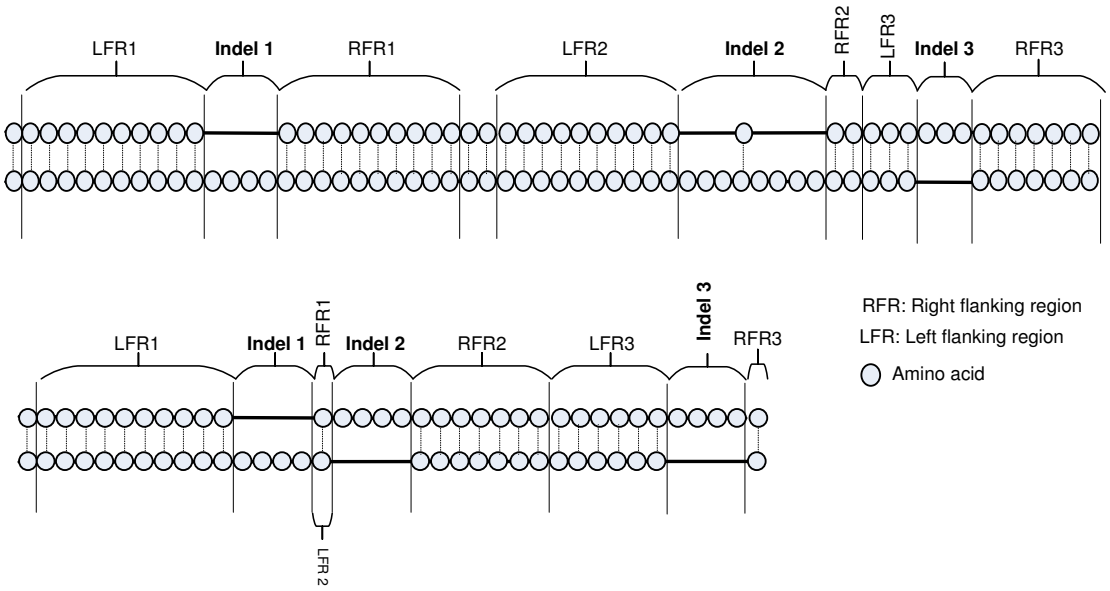


Figure 3.1: Indel regions classified according to the number of amino acids in the flanking regions.

It is to be recalled that in the IndelFR database, a given protein sequence has been aligned with a large number of protein sequences that belong to the same superfamily. For example, consider the protein *d1allb_*. In the IndelFR database, 87 pairwise alignments have been carried out for this protein, and the list of protein names that have been aligned to the protein *d1allb_* are given in Table 3.1. From these alignments, we now identify all the IndelFRs for the protein *d1allb_*, and mark off *IndelFR segments*, which are the segments of the protein sequence to which all

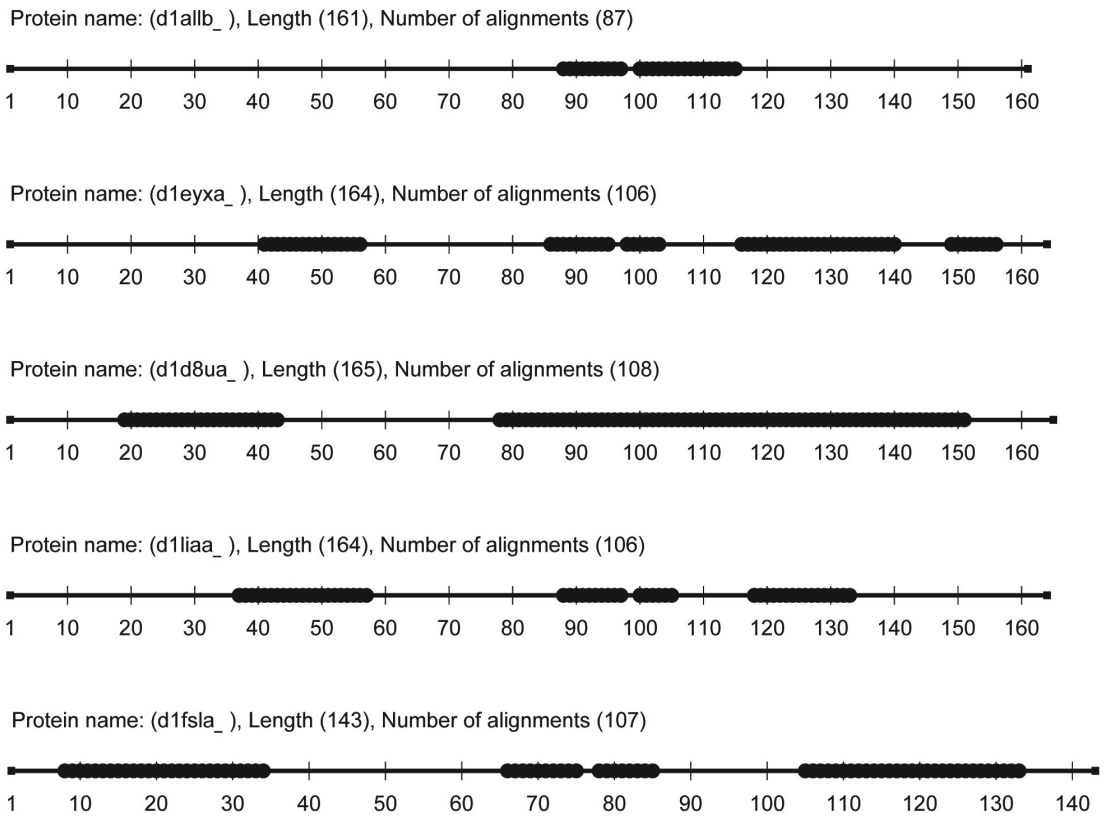


Figure 3.2: IndelFR segments where flanking regions may exist for some selected protein sequences. The segments are indicated by thick lines.

the identified IndelFRs collectively belong to. For the protein sequence *d1allb_*, these segments are observed to be from position 88 to 97 and from 100 to 115, and no indel is located outside these segments. This process can be applied to any of the protein sequences available in the IndelFR database to obtain its IndelFR segments. Figure 3.2 shows such segments for some of the protein sequences selected from the *Globin-like* superfamily, the segments being marked by thick lines.

The results in Figure 3.2 strongly suggest that the IndelFRs for a given protein sequence are conserved within only the IndelFR segments. This is a significant finding, which we will use in Section 3.3 in training the model for the proposed IndelFR predictor.

Table 3.1
Names of proteins that have been aligned to the protein sequence *d1allb_*

d1a4fa_	d1ch4a_	d1g08a_	d1hlma_	d1jl7a_	d1oj6a_	d1spga_
d1a4fb_	d1cqxa1	d1gcva_	d1i3da_	d1la6b_	d1or4a_	d1spgb_
d1a6ma_	d1d8ua_	d1gcvb_	d1idra_	d1lhta_	d1outa_	d1tu9a_
d1a9we_	d1dlwa_	d1gvha1	d1irda_	d1liaa_	d1outb_	d1uc3a_
d1asha_	d1dlya_	d1h97a_	d1irdb_	d1liab_	d1q1fa_	d1urva_
d1b0ba_	d1ecaa_	d1hbra_	d1it2a_	d1mbaa_	d1qpwa_	d1ux8a_
d1b33a_	d1emya_	d1hbrb_	d1itha_	d1mbsa_	d1qpwb_	d1v4wa_
d1b8da_	d1eyxa_	d1hdsa_	d1jeba_	d1mwca_	d1s69a_	d1v4wb_
d1cg5a_	d1fhja_	d1hdsb_	d1jebb_	d1myta_	d1scta_	d1vhba_
d1cg5b_	d1fhjb_	d1hlba_	d1jl6a_	d1ngka_	d1sctb_	d1wmua_
d1wmub_	d1x9fa_	d1x9fb_	d1x9fc_	d1x9fd_	d1xq5a_	d1xq5b_
d2d5xa1	d2d5xb1	d2gdma_	d2h8fa1	d2h8fb1	d2hbga_	d2lhba_
d2mm1a_	d2v1fa1	d3sdha_				

3.3 A Variable–Order Markov Model for Predicting Indel Flanking Reginos

In this section, we present a technique to build an IndelFR predictor for a given protein fold. For this purpose, the protein folds are selected from the following protein classes: *All–α proteins*, *All–β proteins* and *α and β proteins (a/b)*. The selected protein folds from *All–α proteins*, *All–β proteins*, and *α and β proteins (a/b)* are listed in Tables 3.2, 3.3 and 3.4, respectively. More information about the selected protein folds from *All–α proteins*, *All–β proteins* and *α and β proteins (a/b)* are given in Appendix (see Tables A.1, A.2 and A.3). It should be noted that this selection is confined to those protein folds that have indel flanking regions listed in the IndelFR database. It is to be noted that each protein fold contains one or more superfamilies and each superfamily contains one or more protein families. Therefore, the proposed IndelFR predictor for a given protein fold can be used to predict IndelFRs in protein sequences that belong to different protein families within the same fold.

Table 3.2
Protein folds from the *All–α proteins* class that are listed in the IndelFR database

Label	**Number of sequences**	**Protein folds**
A1	109	a.1: Globin–like
A3	50	a.3: Cytochrome c
A4	209	a.4: DNA/RNA–binding 3–helical bundle
A22	29	a.22: Histone–fold
A25	60	a.25: Ferritin–like
A26	11	a.26: 4-helical cytokines
A35	33	a.35: lambda repressor–like DNA–binding domains
A39	102	a.39: EF Hand-like
A45	42	a.45: Glutathione S–transferase (GST), C–terminal domain
A118	68	a.118: alpha–alpha superhelix
A133	47	a.133: Phospholipase A2, PLA2

Since we already know how to obtain the locations of the IndelFR segments for a protein sequence, we build our proposed model for the IndelFR predictor by confining only to the IndelFR segments of each of the protein sequences in a given fold. We extract the flanking regions of all the sequences in the fold and divide them into two sets, the left and right sets. The left set contains all the left flanking regions, whereas the right set contains all the right flanking regions. The flanking regions in either of the two sets have, in general, different lengths, since these lengths, according to our earlier assumption, can vary between one and ten.

The proposed IndelFR predictor for a given protein fold contains two variable–order Markov models (VOMMs) [38,78], one for the left set and the other for the right set. These models learn the conditional probability $P_k(\sigma|s_{i-k}\cdots s_{i-1})$ of observing a particular amino acid $\sigma \in \Psi_{protein}$ at position i, given a context $\mathbf{s} = s_{i-k}\cdots s_{i-1}$ of length k, where the context $\mathbf{s} = s_{i-k}\cdots s_{i-1}$ represents all the previously observed amino acids before σ is observed at position i, each s_j, $j \in \{i-1, \cdots, i-k\}$,

Table 3.3
Protein folds from the *All–β proteins* class that are listed in the IndelFR database

Label	**Number of sequences**	**Protein folds**
B6	62	b.6: Cupredoxin–like
B18	38	b.18: Galactose–binding domain-like
B29	104	b.29: Concanavalin A–like lectins/glucanases
B34	72	b.34: SH3–like barrel
B35	34	Fold b.35: GroES–like
B36	57	b.36: PDZ domain–like
B40	104	b.40: OB–fold
B42	35	b.42: beta–Trefoil
B47	87	b.47: Trypsin–like serine proteases
B50	38	b.50: Acid proteases
B55	63	b.55: PH domain–like
B60	68	b.60: Lipocalins
B82	77	b.82: Double–stranded beta–helix
B121	94	b.121: Nucleoplasmin–like/VP (viral coat and capsid proteins)

representing one of the possible twenty amino acids. The context length k could vary depending on the size and nature of the string of amino acids in a flanking region, and $\Psi_{protein}$ is the alphabet set containing all the amino acid symbols:

$$\Psi_{protein} = \{A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y\}$$

We select a variable–order Markov model instead of a fixed-order one in view of the following reasons: (i) the chosen model should take into consideration varying sizes of the flanking regions in a set, and (ii) it should take care of situations, where a flanking region in which a particular amino acid $\sigma \in \Psi_{protein}$ does not exist for a given context of length m. In the latter case, a VOMM would allow us to reduce the length of the context to be less than m.

In next subsections, we will propose PPM IndelFR predictor using *prediction by partial match* (PPM) [37], and PST IndelFR predictor using *probabilistic suffix tree* (PST) [38].

Table 3.4
Protein folds from the α *and* β *proteins (a/b)* class that are listed in the IndelFR database

Label	**Number of sequences**	**Protein folds**
C1	485	c.1: TIM beta/alpha–barrel
C2	288	c.2: NAD(P)–binding Rossmann-fold domains
C3	61	c.3: FAD/NAD(P)–binding domain
C14	29	c.14: ClpP/crotonase
C23	117	c.23: Flavodoxin–like
C26	69	c.26: Adenine nucleotide alpha hydrolase–like
C36	29	c.36: Thiamin diphosphate–binding fold (THDP–binding)
C37	340	c.37: P-loop containing nucleoside triphosphate hydrolases
C47	158	c.47: Thioredoxin fold
C55	122	c.55: Ribonuclease H–like motif
C56	59	c.56: Phosphorylase/hydrolase-like
C61	36	c.61: PRTase–like
C67	89	c.67: PLP–dependent transferase–like
C68	38	c.68: Nucleotide-diphospho-sugar transferases
C69	100	c.69: alpha/beta–Hydrolases
C94	71	c.94: Periplasmic binding protein–like II
C95	24	c.95: Thiolase–like
C108	54	c.108: HAD–like

3.3.1 PPM IndelFR Predictor

In PPM, in order to build a VOMM for the left (right) set of flanking regions, we start by analyzing a subset of the left (right) flanking regions as the training set and counting the number of occurrences of the amino acid σ immediately after the context $\mathbf{s}$, that is, counting the number of occurrences of the pattern $\mathbf{s}\sigma$ in the training set for each amino acid $\sigma \in \Psi_{protein}$ and for each context $\mathbf{s} = s_{i-k} \cdots s_{i-1}$ of length k, where each s_j, $j \in \{i-1, \cdots, i-k\}$, represents one of the twenty possible amino acids. The context length k varies from zero to D, where D is the memory length of the VOMM. Hence, for each value of k, we can compute the conditional empirical probability $\widetilde{P}_k(\sigma|\mathbf{s})$ [1] as

$$\widetilde{P}_k(\sigma|\mathbf{s}) = \frac{N_{\mathbf{s}\sigma}}{\sum\limits_{s_j \in \Psi_{protein}} N_{\mathbf{s}s_j}} \tag{3.1}$$

where $N_{\mathbf{s}\sigma}$ is the number of occurrences of the pattern $\mathbf{s}\sigma$ in the training set. For $k = 0$, we can calculate the conditional empirical probability, $\widetilde{P}(\sigma|\epsilon)$, where ϵ represents an empty context. PPM handles the zero frequency problem by going through the mechanisms of *escape* and *exclusion* [1]. In the escape mechanism, for each context $\mathbf{s} = s_{i-k} \cdots s_{i-1}$ of length k, we make use of a probability mass $P_k(escape|\mathbf{s})$ for all the amino acids that do not appear after the context $\mathbf{s} = s_{i-k} \cdots s_{i-1}$ in the training set. There are different ways of defining the escape probabilities for a context. These definitions are generally based on intuition and experience, and not on any underlying theory. For example, in [79], this escape probability has been defined as

$$P_k(escape|\mathbf{s}) = \frac{|\Psi_{\mathbf{s}}|}{|\Psi_{\mathbf{s}}| + \sum\limits_{s_j \in \Psi_{\mathbf{s}}} N_{\mathbf{s}s_j}} \tag{3.2}$$

where $\Psi_{\mathbf{s}}$ is a set of amino acids appearing after the context $\mathbf{s} = s_{i-k} \cdots s_{i-1}$, i.e., $\Psi_{\mathbf{s}} = \{\sigma : N_{\mathbf{s}\sigma} > 0\}$, and $|\Psi_{\mathbf{s}}|$ denotes the number of elements in $\Psi_{\mathbf{s}}$. Accordingly,

the conditional probability is modified as

$$P_k(\sigma|\mathbf{s}) = \begin{cases} P_k(\sigma|\mathbf{s}) & \text{if } \sigma \in \Psi_{\mathbf{s}} \\ P_k(escape|\mathbf{s}) P_{k-1}(\sigma|s_{i-k+1} \cdots s_{i-1}) & otherwise \end{cases} \tag{3.3}$$

where

$$P_k(\sigma|\mathbf{s}) = \frac{N_{\mathbf{s}\sigma}}{|\Psi_{\mathbf{s}}| + \sum_{s_j \in \Psi_{protein}} N_{\mathbf{s}s_j}} \tag{3.4}$$

In the above equation, if $P_{k-1}(\sigma|s_{i-k+1} \cdots s_{i-1})$ is zero, then Equation (3.3) is recursively modified by using contexts of shorter lengths:

$$P_{k-1}(\sigma|s_{i-k+1} \cdots s_{i-1}) = \Big(P_{k-1}(escape|s_{i-k+1} \cdots s_{i-1}) \; P_{k-2}(\sigma|s_{i-k+2} \cdots s_{i-1}) \Big) \tag{3.5}$$

In the exclusion mechanism, if a prediction fails for a certain context, then the unseen amino acid cannot be one of the amino acids that has been observed after that context, and the relevant alphabet set for all the shorter contexts should be reduced by eliminating these observed amino acids. Hence, every amino acid $\sigma \in \Psi_{\mathbf{s}}$ observed after context $\mathbf{s} = s_{i-k} \cdots s_{i-1}$ is excluded, when we calculate the conditional probability for all contexts shorter than $\mathbf{s} = s_{i-k} \cdots s_{i-1}$. For the purpose of illustration, an example showing how the PPM structure of VOMM can be constructed from the set of flanking regions is given below.

Example 2.

Given the training set {ABEACADABE, ACAABED, ADACADED, EDABAC, CACAED, DABACAE, ABADEDCC, ABAC} of flanking regions, the PPM structure with $D = 2$, and alphabet set $\Psi_{protein} = \{A, B, C, D, E\}$ can be contracted as given by Table 3.5. This table lists the frequencies of different characters following the different contexts including the empty context.

Table 3.5
PPM using the training set of flanking regions {ABEACADABE, ACAABED, ADACADED, EDABAC, CACAED, DABACAE, ABADEDCC, ABAC}, with $D = 2$, and alphabet set $\Psi_{protein} = \{A, B, C, D, E\}$.

Order k	Context	Character frequencies					Total	Escape counts
		A	B	C	D	E		
2	ED	1		1			**2**	2
2	EA			1			**1**	1
2	DE				2		**2**	1
2	DC			1			**1**	1
2	DA		3	1			**4**	2
2	CA	1		1	2	2	**6**	4
2	BE	1			1		**2**	2
2	BA			3	1		**4**	2
2	AE				1		**1**	1
2	AD	2				2	**4**	2
2	AC	5					**5**	1
2	AB	4				3	**7**	2
2	AA		2				**2**	1
1	E	1			5		**6**	2
1	D	4		1		2	**7**	3
1	C	6		1			**7**	2
1	B	4				3	**7**	2
1	A	2	7	7	4	2	**22**	5
0	ϵ	22	7	10	10	8	**57**	5

■

3.3.2 PST IndelFR Predictor

In the PST structure, a single tree with depth D is constructed to represent a VOMM of memory length D. The nodes in the tree have different degrees varying from zero (for leaves) to the size of the alphabet set $\Psi_{protein}$ (for the internal nodes and the root). Each edge in the tree is labeled by a single amino acid from the set $\Psi_{protein}$. Each node in the tree is labeled by a unique context $\mathbf{s} = s_{i-k} \cdots s_{i-1}$, where the context length varies from zero (for the root) to D. Also, each node is assigned a conditional probability $P_k(\sigma|\mathbf{s})$, where $\sigma \in \Psi_{protein}$. It should be noted that the context $\mathbf{s} = s_{i-k} \cdots s_{i-1}$ is generated by moving from the node to the root (i.e., in PST, the father of the node labeled by $s_1 s_2 s_3$ is the node $s_2 s_3$, and not $s_1 s_2$ as in a regular suffix tree).

To build the PST structure $\mathbf{T}$ for the left or for the right flanking region, we need to set the following four parameters:

i. The memory length parameter D of PST.

ii. The context threshold parameter N_T, where $N_T = c \cdot m$, c $(0 < c < 1)$ being a constant and m the total number of flanking regions in the left or right set. The parameter N_T determines as to which contexts would be included in building the PST structure. If the number of occurrences of a context is less than N_T, then such a context is excluded in building the structure.

iii. The parameter r is used to determine whether the context $\mathbf{s} = s_{i-k} \cdots s_{i-1}$ contributes additional information in predicting the amino acid σ relative to its "parent" or "suffix" context $s_{i-k+1} \cdots s_{i-1}$, denoted by $suf(\mathbf{s})$. The ratio $P_k(\sigma|\mathbf{s})/P_{k-1}(\sigma|suf(\mathbf{s}))$ is chosen to be outside the interval $(1/r, r)$. In order to make the contribution of this context to be sensitive, r is chosen to be $1+\delta$, δ being a small quantity.

iv. The parameter B_s is chosen to be $B_s = 5 \cdot |\Psi_s|$, as suggested in [80], in order

to ensure that for a given context the probability of an amino acid $\sigma \in \Psi_{protein}$ does not become zero.

Let N_s be the number of occurrences of the context $\mathbf{s} = s_{i-k} \cdots s_{i-1}$ in the training set of the left (right) flanking regions, $N_{\mathbf{s}\sigma}$ the number of occurrences of the pattern $\mathbf{s}\sigma$ in the left (right) training set, and $P_k(\sigma|\mathbf{s})$ the conditional probability associated with the node labeled by the context $\mathbf{s} = s_{i-k} \cdots s_{i-1}$. The various steps to build the PST structure for the left (right) training set are as follows.

Step 1: Create a tree **T** with a single root node labeled by an empty context ϵ, and create an empty set $\mathbf{set}_{ptr}$.

Step 2: Add to the set $\mathbf{set}_{ptr}$ all the contexts of length unity that have occurred more number of times than N_T, the context threshold (i.e., $N_s > N_T$).

$$\mathbf{set}_{ptr} \leftarrow \{\mathbf{s}|N_{\mathbf{s}} > N_T\}$$

Step 3: Select a context from $\mathbf{set}_{ptr}$.

Step 4: Test if there is an amino acid $\sigma \in \Psi_s$ that has a conditional empirical probability $\widetilde{P}_k(\sigma|\mathbf{s})$ given by Equation (3.1) satisfying the inequality:

$$\widetilde{P}_k(\sigma|\mathbf{s}) > \frac{1}{|\Psi_{protein}|}$$

Step 5: Test if there is an amino acid σ (not necessarily the same amino acid as in Step 4) from alphabet Ψ_s satisfying the condition

$$\frac{\widetilde{P}_k(\sigma|\mathbf{s})}{\widetilde{P}_{k-1}(\sigma|suf(\mathbf{s}))} = \begin{cases} \geq r \\ or \\ \leq \frac{1}{r} \end{cases}$$

Step 6: If the conditions in Steps 4 and 5 are both satisfied, go to Step 7; otherwise

(i.e. condition in Step 4 or Step 5 is not satisfied), go to Step 10.

Step 7: Test if the parent node of the context **s**, labelled by $suf(\mathbf{s})$, already exists in the tree **T**. If yes, add the node corresponding to this context **s** to the tree **T**.

Step 8: If the parent node for **s** in Step 7 does not exist, then create a node for this context **s** and for its parent node. If the parent node of the latter does not exist, then repeat this procedure until an existing parent node in the tree is reached.

As an example, to illustrate this step, assume a node labeled by context $\mathbf{s} = C$ exists in the tree, and we are trying to add a node labeled by context $\mathbf{s} = ABDC$ to the tree. In PST, the parent node for the context $\mathbf{s} = ABDC$ is a node labeled by context BDC, which does not exist in the tree. Also, the parent node of the context BDC is a node labeled by context DC, which also does not exist in the tree. But the parent node of the context DC, namely, the node labeled C exists in the tree. Hence, we have to add two more nodes labeled BDC and DC, in addition to the node labeled $ABDC$ to the tree. This is illustrated in the Figure 3.3.

Step 9: Adjust the conditional probability for each added node in Step 7 or 8, so that the probability of an amino acid $\sigma \in \Psi_{protein}$ for a given context is given by

$$P_k(\sigma|\mathbf{s}) = \frac{N_{\mathbf{s}\sigma} + \left(\frac{1}{|\Psi_{protein}|}\right)B_s}{\sum\limits_{s_j \in \Psi_s} N_{\mathbf{s}s_j} + B_s}$$

Step 10: If the length of $\mathbf{s} < D$, and there exists a pattern $\sigma\mathbf{s}$, which has occurred more number of times than N_T (i.e.,$N_{\sigma\mathbf{s}} \geq N_T$), then add the pattern $\sigma\mathbf{s}$ to $\mathbf{set}_{ptr}$:

$$\mathbf{set}_{ptr} \leftarrow \{\sigma\mathbf{s}|\sigma \in \Psi_{protein} \textbf{ and } N_{\sigma\mathbf{s}} \geq N_T\}$$

Step 11: Remove the context **s** from $\mathbf{set}_{ptr}$, and repeat Steps 3 to 11 until $\mathbf{set}_{ptr}$ becomes empty.

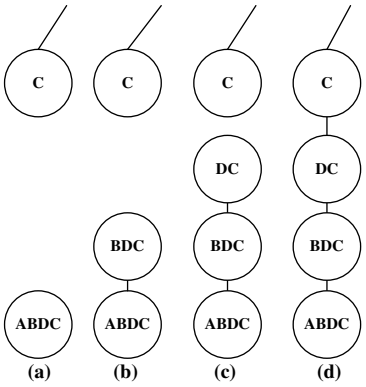


Figure 3.3: An illustration for Step 8 of the procedure used to build the PST structure.

Example 3 given below illustrates how a PST structure with the parameters $D = 2, N_T = 0.01$ and $r = 1.05$ can be built from a set of flanking regions.

Example 3.

Given the training set {ABEACADABE, ACAABED, ADACADED, EDABAC, CACAED, DABACAE, ABADEDCC, ABAC} of flanking regions, the PST structure with the parameters $D = 2, N_T = 0.01$, $r = 1.05$ and the alphabet set $\Psi_{protein} = \{A, B, C, D, E\}$ can be constructed as shown in Figure 3.4.

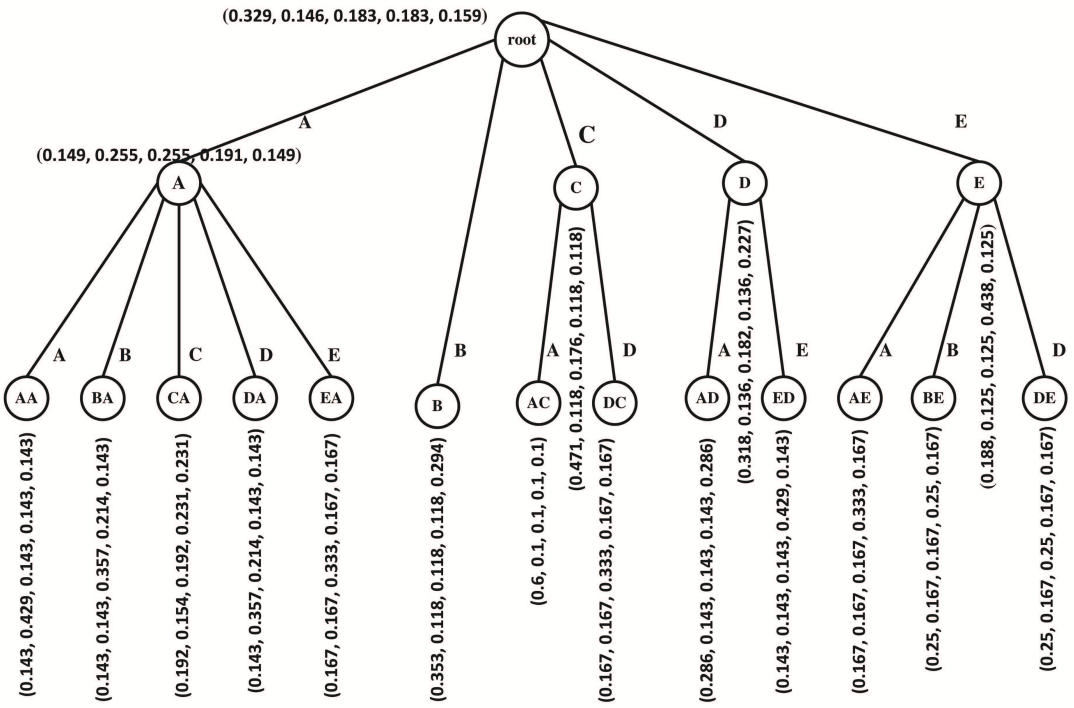


Figure 3.4: The PST structure using the training flanking regions {ABEACADABE, ACAABED, ADACADED, EDABAC, CACAED, DABACAE, ABADEDCC, ABAC}, with the parameters $D = 2, N_T = 0.01$, $r = 1.05$ and the alphabet set $\Psi_{protein} = \{A, B, C, D, E\}$. Each node is labelled with a context $\mathbf{s}$ and has conditional probability $(P(A|\mathbf{s}) , P(B|\mathbf{s}) , P(C|\mathbf{s}), P(D|\mathbf{s}) , P(E|\mathbf{s}))$.

■

3.4 Prediction of Indel flanking regions using VOMM

Given a test protein sequence $\mathbf{S}^n = s_1 s_2 s_3 \cdots s_n$ of length n, we scan it using a running window of length L moving it one amino acid at a time. To determine whether or not the string of amino acids within a window contains a flanking region, we compute the probability of this string using VOMM.

For a VOMM, we use $P(\mathbf{win}_i)$ to denote the probability of the string $\mathbf{seg}_i = s_i s_{i+1} \cdots s_{i+L-1}$ of length L. If VOMM has a memory length $D < L$, then $P(\mathbf{win}_i)$, which is also referred to as the likelihood of $\mathbf{win}_i$, is given by [1]

$$\begin{aligned} P(\mathbf{win}_i) =& P_0(s_i) P_1(s_{i+1}|s_i) \cdots \\ & P_D(s_{i+L-1}|s_{i+L-D} \cdots s_{i+L-2}) \end{aligned} \tag{3.6}$$

We calculate the various probabilities on the right side of Equation (3.6) by using PPM or PST. If the probability $P_k(s_j|s_{j-k}s_{j-k+1}\cdots s_{j-1})$ for s_j, $(i \le j \le i+L-1)$ does not exist, we proceed as follows.

a) In the case of PPM, we use the escape and exclusion mechanisms, in conjunction with Equation (3.3) to calculate each of the probabilities in Equation (3.6).

b) In the case of PST, we find the longest suffix of the context $s_{j-k}s_{j-k+1}\cdots s_{j-1}$ that exists in the tree. Assuming the longest suffix of the context $s_{j-k}s_{j-k+1}\cdots s_{j-1}$ that exists in the tree to be $s_{j-t}s_{j-t+1}\cdots s_{j-1}$, $(0 \le t < k)$, then

$$P_k(s_j|s_{j-k}s_{j-k+1}\cdots s_{j-1}) = P_t(s_j|s_{j-t}s_{j-t+1}\cdots s_{j-1}) \tag{3.7}$$

Maximizing the likelihood $P(\mathbf{win}_i)$ is equivalent to minimizing the *average log-*

loss function [1] defined as

$$
\begin{aligned}
loglossP(\mathbf{win}_i) = -\frac{1}{L}\Bigg(& \log P_0(s_i) + \log P_1(s_{i+1}|s_i) + \\
& \log P_2(s_{i+2}|s_1 s_{i+1}) + \cdots + \\
& \log P_D(s_{i+L-1}|s_{i+L-1-D}\cdots s_{i+L-2})\Bigg)
\end{aligned}
\tag{3.8}
$$

where the logarithm taken with a base of 2. Hence, $\mathbf{win}_i$ contains a flanking region if it has a low average log–loss value compared to that of its neighboring windows. Example 4 below illustrates the steps for calculating the probability of a particular segment using PPM or PST.

Example 4.

For the training set {ABEACADABE, ACAABED, ADACADED, EDABAC, CACAED, DABACAE, ABADEDCC, ABAC} of flanking regions, we constructed PPM and PST structures in Example 2 and Example 3, respectively. Using Table 3.5 of Example 2, we can calculate $P(BAABEDCA)$ as

$$
\begin{aligned}
P(BAABEDCA) = \Bigg(& P_0(B|\epsilon)P_1(A|B)P_2(A|BA) \\
& P_2(B|AA)P_2(E|AB)P_2(D|BE) \\
& P_2(C|ED)P_2(A|DC)\Bigg)
\end{aligned}
$$

$$
\begin{aligned}
P(BAABEDCA) = & \Bigg(\Big(\frac{7}{57+5}\Big)\Big(\frac{4}{7+2}\Big)\Big((\frac{2}{4+2})(\frac{2}{11+5})\Big) \\
& \Big(\frac{2}{2+1}\Big)\Big(\frac{3}{7+2}\Big)\Big(\frac{1}{2+2}\Big) \\
& \Big(\frac{1}{2+2}\Big)\Big((\frac{1}{1+1})(\frac{6}{6+2})\Big)\Bigg) \\
& = 1.089 \times 10^{-5}
\end{aligned}
$$

and, we can calculate $P(BAABEDCA)$ using Figure 3.4 of Example 3 as

$$
\begin{aligned}
P(BAABEDCA) = & \Bigg(P_0(B|\epsilon)P_1(A|B)P_2(A|BA) \\
& P_2(B|AA)P_2(E|AB)P_2(D|BE) \\
& P_2(C|ED)P_2(A|DC)\Bigg)
\end{aligned}
$$

$$
\begin{aligned}
P(BAABEDCA) = & \Bigg(\Big(0.146\Big)\Big(0.353\Big)\Big(0.143\Big) \\
& \Big(0.429\Big)\Big(0.294\Big)\Big(0.25\Big) \\
& \Big(0.25\Big)\Big(0.167\Big)\Bigg) \\
& = 9.702 \times 10^{-6}
\end{aligned}
$$

■

The proposed IndelFR predictor for a given protein fold can be built using PPM or PST. We build the left PPM (LPPM) and the left PST (LPST) for the left set, and build the right PPM (RPPM) and the right PST (RPST) for the right set. LPPM and RPPM are combined together to form a *PPM IndelFR predictor* for memory length D. Similarly, LPST and RPST are combined together to form a *PST IndelFR*

predictor for memory length D. Such IndelFR predictors are built for various values of the memory length D, $D \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, so that we can determine the value of D that results in the best performance in predicting the locations of IndelFRs. The procedure to extract the predicted locations of these regions in the test protein sequence using PPM IndelFR predictor is given in Algorithm 1. A similar procedure is applied for the proposed PST IndelFR predictor.

Algorithm 1: Procedure to extract the predicted locations of IndelFRs in a test protein sequence using the proposed PPM IndelFR predictor with a memory length D.

Step 1: Scan the test protein sequence $\mathbf{S}^n = s_1 s_2 s_3 \cdots s_n$ of length n using a running window of length $L = 10$.

$$\mathbf{win}_i = s_i s_{i+1} \cdots s_{i+9}, \quad 1 \leq i \leq (n-9)$$

Step 2: Compute and store the average log-loss values for each window using LPPM and RPPM with a memory length D using Equation (3.8).

Step 3: From the LPPM average log-loss values, choose the mean of these values as the threshold. Then, find the locations of the local minima that have values below the threshold.

Step 4: Repeat Step 3 using the RPPM log-loss values and find the locations of the local minima.

Step 5: Find the locations of IndelFRs in the test protein sequence by identifying each of the LPPM minimum locations that is immediately followed by a RPPM minimum location. The identified LPPM and the corresponding RPPM minimum locations represent the start locations of the predicted left and right flanking regions, respectively, and each flanking region (left or right) has a length of at most 10. For each selected minimum location at ν, the predicted locations for this flanking region (left or right) are limited to $\nu, \nu+1, \nu+2, \cdots, \nu+9$.

Step 6: For the test protein sequence, use the actual locations of IndelFRs and the predicted locations to determine the *accuracy* and the F_1-*measure*.

3.5 Results and Discussion

In this section, we evaluate the performance of the proposed PPM and PST IndelFR predictors. For this purpose, we select from SCOP database 11, 14 and 18 protein folds from different protein classes: *All-α proteins*, *All-β proteins*, and *α and β proteins (a/b)*, respectively. The selected protein folds are listed in Tables 3.2, 3.3 and 3.4, respectively. This selection is confined to those protein folds that have indel flanking regions listed in the IndelFR database. The proposed PPM and PST IndelFR predictors are built for each of the selected protein folds. These predictors are built for various values of the memory length D, $D \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, so that we can determine the value of D that provides the best performance.

We use the k-fold cross-validation method for training and testing the proposed IndelFR predictors, where $k = 10$. Consequently, k iterations of training and testing are performed for each predictor. In the training phase, we train the IndelFR predictor for a given protein fold using the indel flanking regions listed in the IndelFR database. In the testing phase, we first test the trained predictors on the protein sequences from the same protein fold belonging to the IndelFR database and next, on the set of protein sequences from the same protein fold but belonging to the sequence alignment benchmark (SABmark 1.65) [39]. Finally, the performance of the two proposed predictors is compared to that using the latest version of the alignment software HMMER, HMMER 3.0 [40].

We evaluate the performance of the proposed predictors using the measures of *accuracy* and F_1*–measure*, which are the commonly–used metrics in the evaluation of the performance of prediction techniques in bioinformatics [81–83].

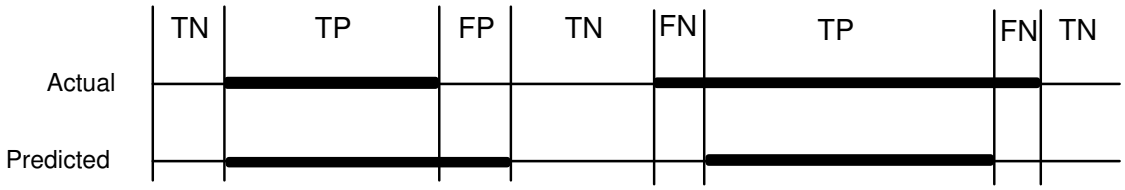


Figure 3.5: An illustration for determining the four quantities TP, FN, TN and TN for a given protein sequence, where the thick lines indicate the various flanking segments.

3.5.1 Performance evaluation using accuracy and the F_1–measure

The *accuracy* and F_1*–measure* are the commonly–used metrics in the evaluation of the performance of prediction techniques in bioinformatics [81–83]. The *accuracy* is the ratio of correct predictions to the total number of predictions

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{3.9}$$

where

TP is the number of amino acids that are located in the various flanking segments for a given protein sequence and have been so predicted,

FP the number of amino acids that are not located in any of the flanking segments, but have been predicted to be in the flanking segments,

FN the number of amino acids that are located in the various flanking segments, but have not been so predicted, and

FN the number of amino acids that are not located in any of the flanking segment and have been so predicted.

An illustration as to how TP,FN, TN and TN are determined is shown in Figure 3.5.

The F_1–*measure* is defined as [83]

$$F_1 - measure = \frac{2PR}{P+R} \tag{3.10}$$

where P is the *precision* (*selectivity*) and R is the *recall* given by

$$\begin{aligned} P &= \frac{TP}{TP+FP} \\ R &= \frac{TP}{TP+FN} \end{aligned} \tag{3.11}$$

The F_1–*measure* has a value in the range $[0, 1]$.

3.5.2 Prediction in IndelFR database

The average log–loss values for each test protein sequence is computed for each of the two proposed predictors. For the purpose of illustration, the average log–loss values using the two predictors for the protein sequence *d1liab_* are shown in Figure 3.6. It is seen from this figure that the average log–loss values around a flanking region are very much less than those around the other regions. It should be noted that for each of the test protein sequences, we follow the steps outlined in Algorithm 1 to extract the predicted locations of IndelFRs, and to calculate both the accuracy and the F_1–measure. As seen from Figure 3.6(a), the PPM IndelFR predictor predicts the locations (A, B), (C, D), (E, F), and (G, H) as the start locations for IndelFRs (left and right, respectively), while this predictor ignores the RPPM minimum location (I) as it is not preceded by an LPPM minimum location. To compute the accuracy and the F_1–measure, we use the actual IndelFRs shown in Figure 3.6(c) taken from the IndelFR database, and the locations predicted by the proposed PPM IndelFR predictor. The accuracy and F_1–measure are found to be 81% and 77%, respectively. In a similar manner, using the results shown

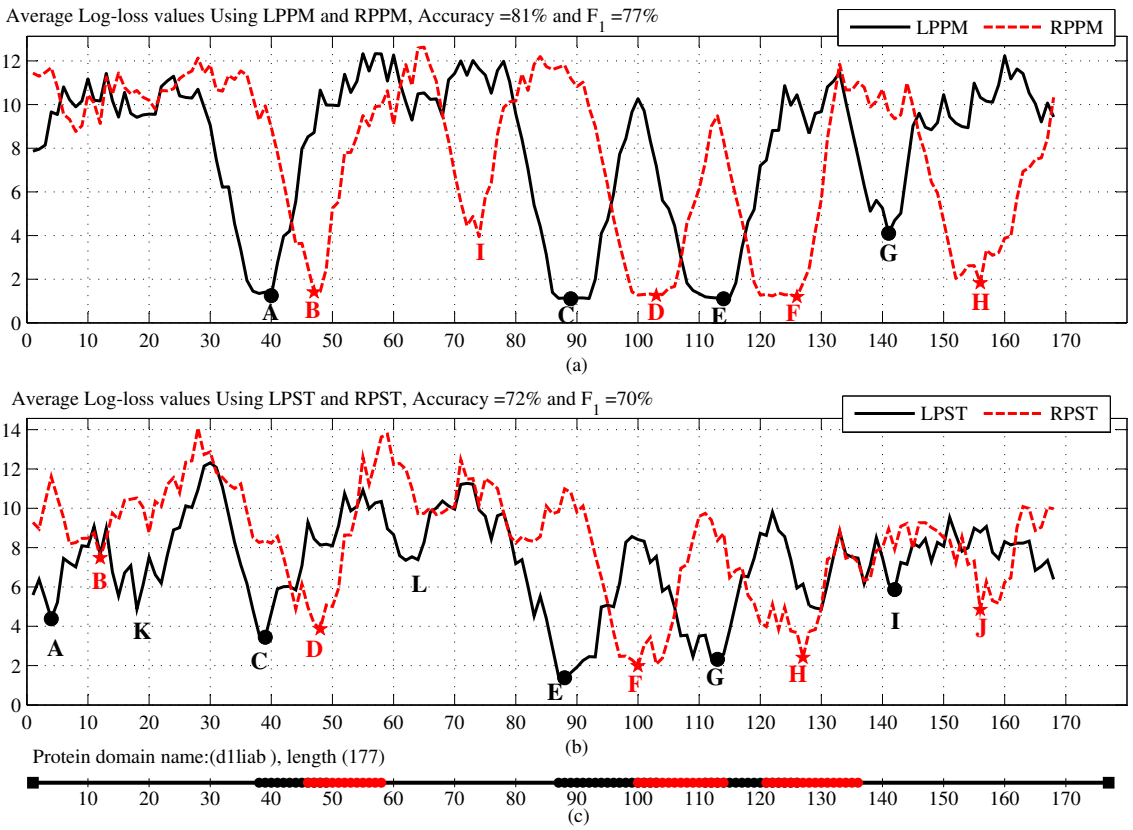
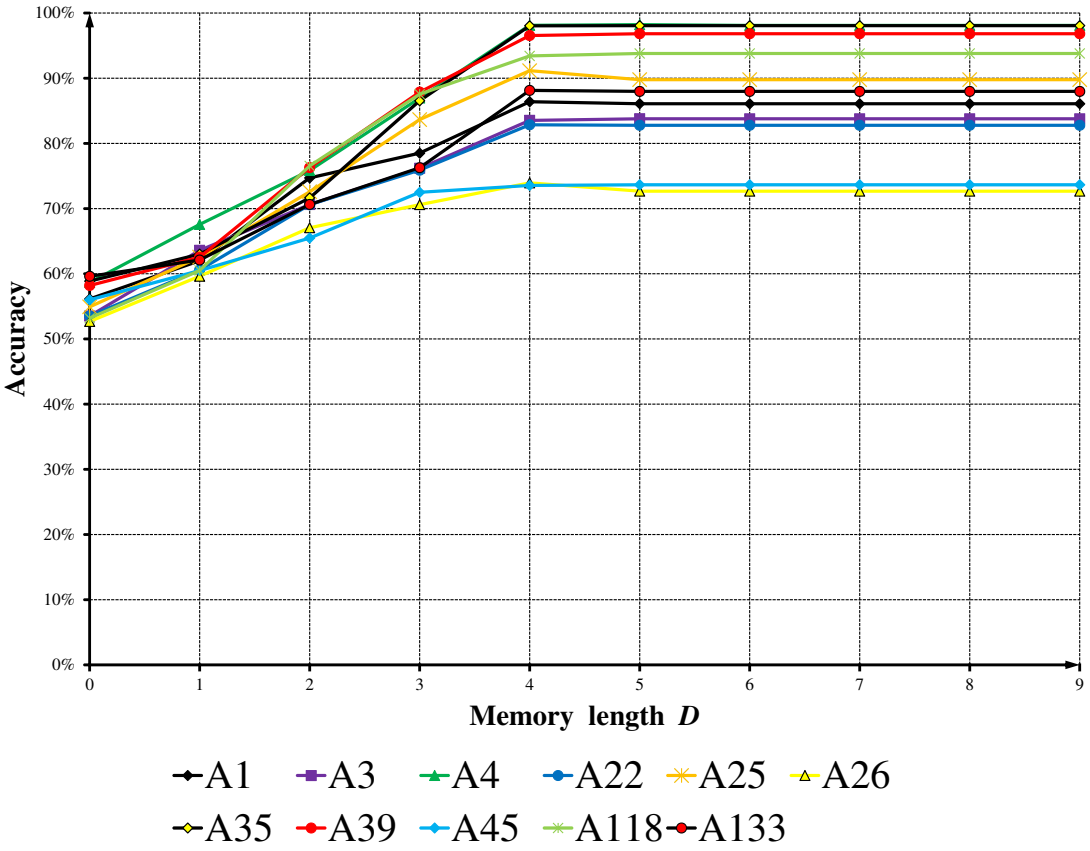


Figure 3.6: Average log-loss values for the *d1liab_* protein sequence using (a) PPM IndelFR predictor, and (b) PST IndelFR predictor. (c) Ground truth for the IndelFR taken from the IndelFR database [17]. Solid dots represent the start locations of the predicted left flanking regions and the stars that of the predicted right flanking regions.

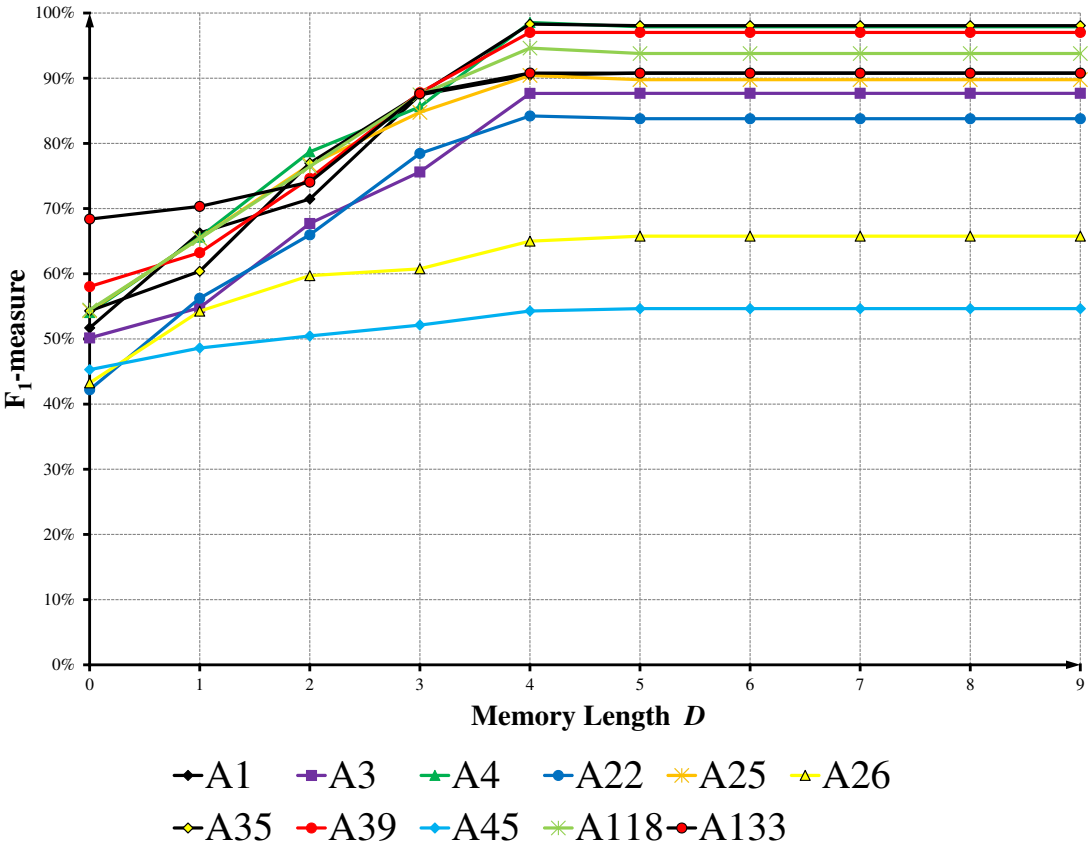
in Figure 3.6(b), we determine the accuracy and F_1–measure for the PST IndelFR predictor to be 72% and 70%, respectively.

The average accuracy and F_1–measure values of the PPM predictor for various values of the memory length D, $D \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, are obtained for each of the 11 chosen protein folds from the *All-α protein* class, and are shown in Figures 3.7 and 3.8, respectively. It is observed from these figures that the best choice for the memory length D is 4. Further, the accuracy varies from 74% to 98% and the F_1–measure from 54% to 99% for the various folds. The average accuracy and F_1–measure values of the PPM predictor, with a memory length of 4, over the *All-α protein* class are 91% and 92%, respectively. In a similar manner, the average accuracy and F_1–measure values of the PST predictor for various values of the memory length D, $D \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, are obtained for each the 11 chosen protein folds from the *All-α protein* class , and are shown in Figures 3.9 and 3.10, respectively. The results for the PST predictor strongly suggest that the best choice for the memory length D is again 4. Further, the accuracy varies from 63% to 96% and the F_1–measure from 54% to 97% for the various folds. The average accuracy and F_1-measure values of the PST predictor, with a memory length of 4, over the *All-α protein* class are 88% and 89%, respectively.



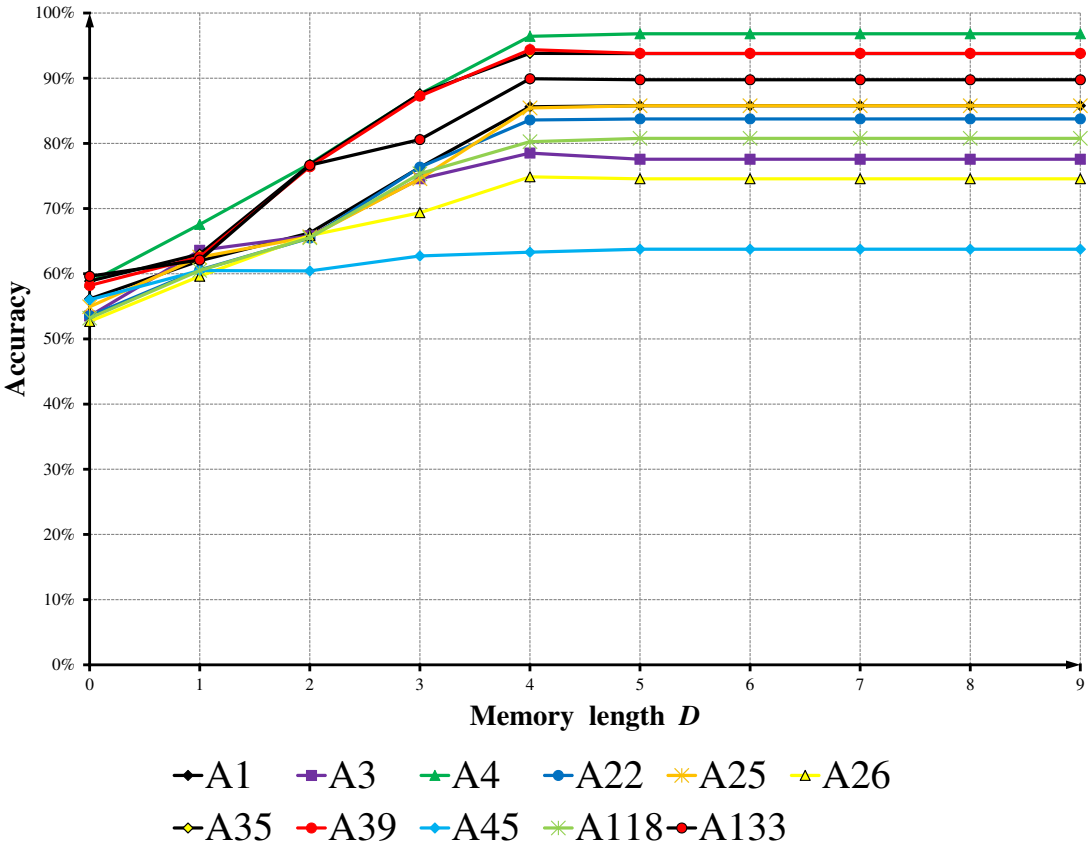
Note: A1, A3, A4, A22, A25, A26, A35, A39, A45, A118 and A133 are the protein folds given in Table 3.2

Figure 3.7: Average values of *accuracy* for the PPM IndelFR predictor for different protein folds selected from the *All-α protein* class for various values of the memory length D.



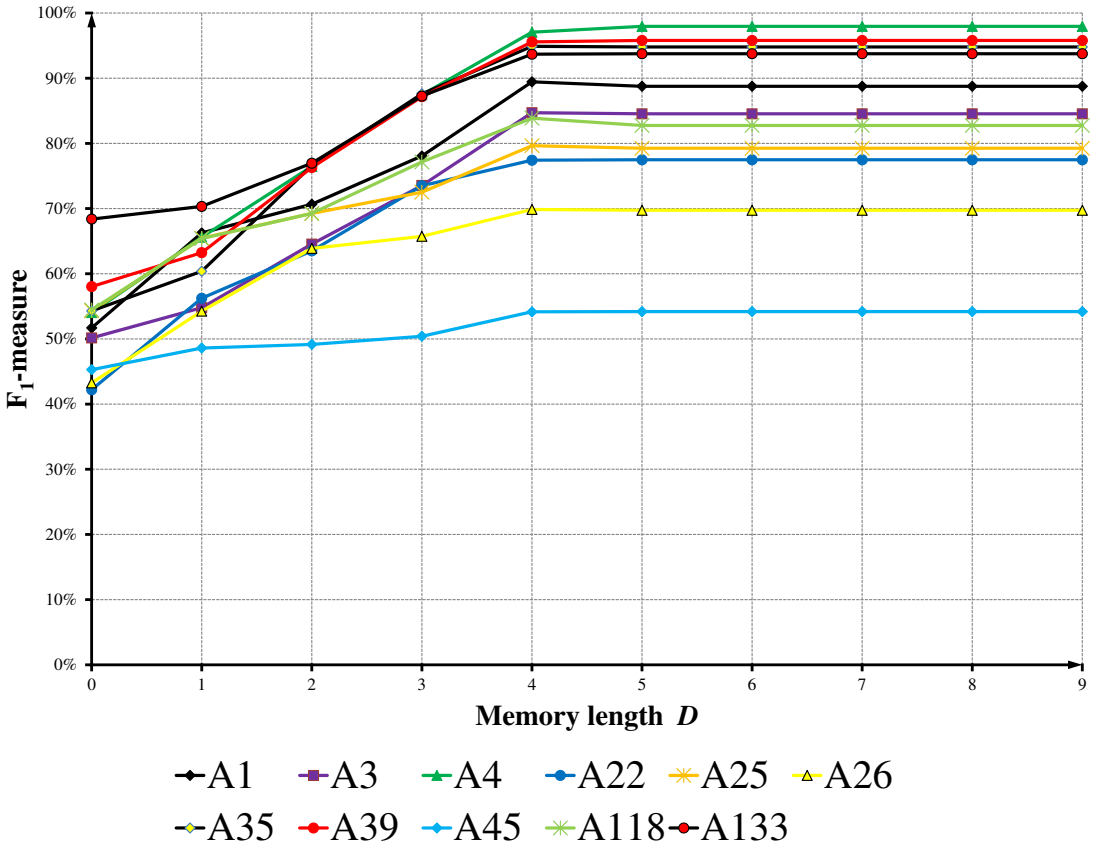
Note: A1, A3, A4, A22, A25, A26, A35, A39, A45, A118 and A133 are the protein folds given in Table 3.2

Figure 3.8: Average values of F_1–*measure* for the PPM IndelFR predictor for different protein folds selected from the *All-α protein* class for various values of the memory length D.



Note: A1, A3, A4, A22, A25, A26, A35, A39, A45, A118 and A133 are the protein folds given in Table 3.2

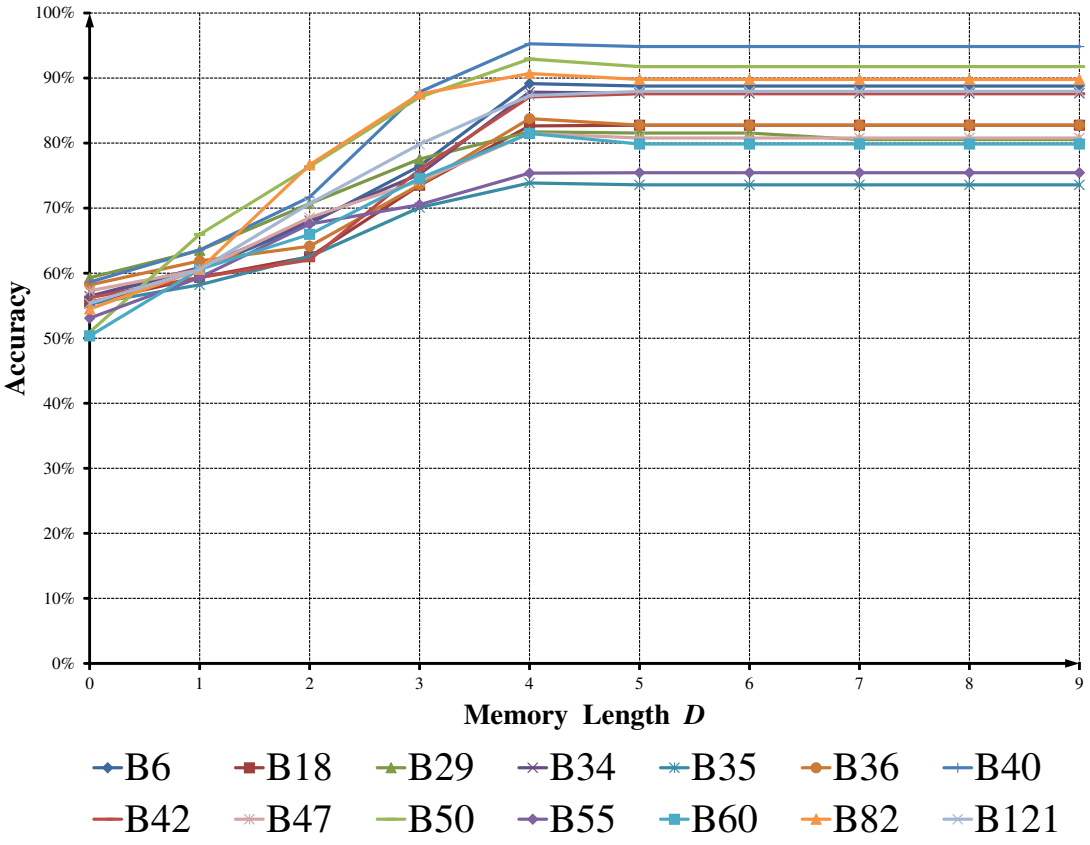
Figure 3.9: Average values of *accuracy* for the PST IndelFR predictor for different protein folds selected from the *All-α protein* class for various values of the memory length D.



Note: A1, A3, A4, A22, A25, A26, A35, A39, A45, A118 and A133 are the protein folds given in Table 3.2

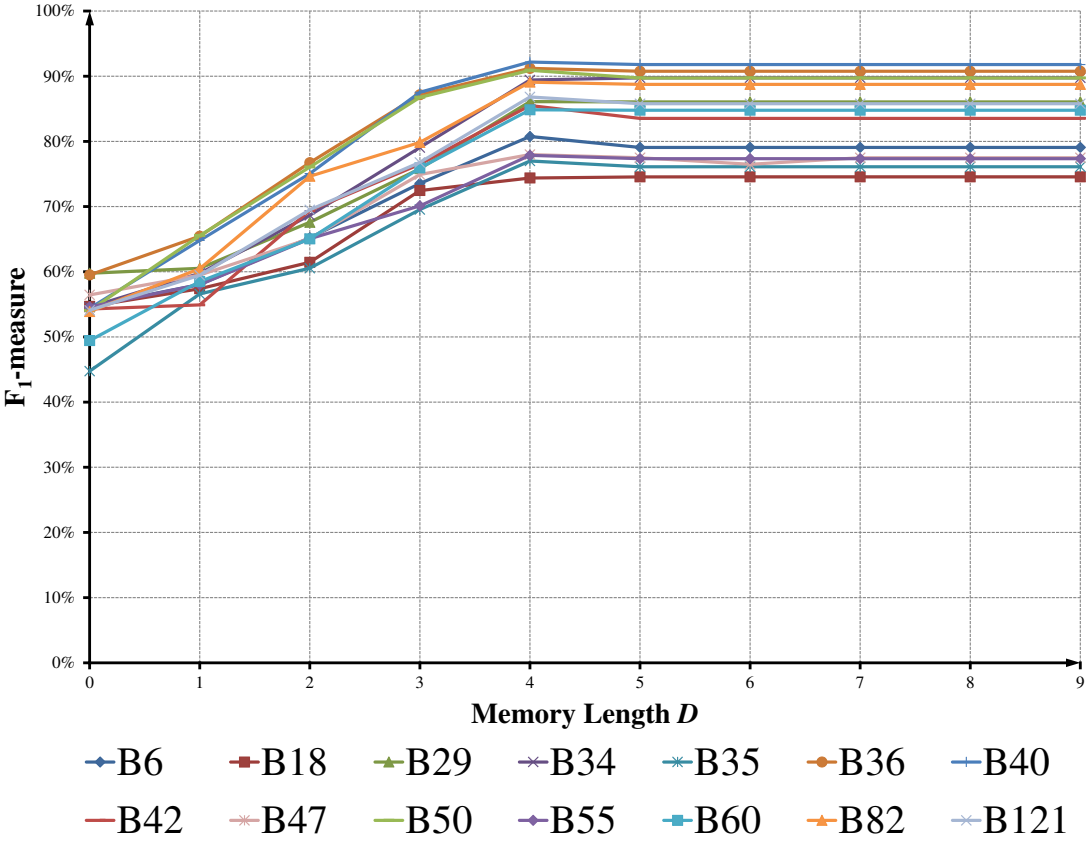
Figure 3.10: Average values of F_1–*measure* for the PST IndelFR predictor for different protein folds selected from the *All-α protein* class for various values of the memory length D.

The average accuracy and F_1–measure values of the PPM predictor for various values of the memory length D, $D \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, are obtained for each of the 14 chosen protein folds from the *All-β protein* class, and are shown in Figures 3.11 and 3.12, respectively. It is observed from these figures that the best choice for the memory length D is 4. Further, the accuracy varies from 74% to 95% and the F_1–measure from 74% to 92% for the various folds. The average accuracy and F_1–measure values of the PPM predictor, with a memory length of 4, over the *All-β protein* class are 86% and 85%, respectively. In a similar manner, the average accuracy and F_1–measure values of the PST predictor for various values of the memory length D, $D \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, are obtained for each the 14 chosen protein folds from the *All-β protein* class , and are shown in Figures 3.13 and 3.14, respectively. The results for the PST predictor strongly suggest that the best choice for the memory length D is again 4. Further, the accuracy varies from 74% to 97% and the F_1–measure from 76% to 97% for the various folds. The average accuracy and F_1-measure values of the PST predictor, with a memory length of 4, over the *All-β protein* class are 84% and 86%, respectively.



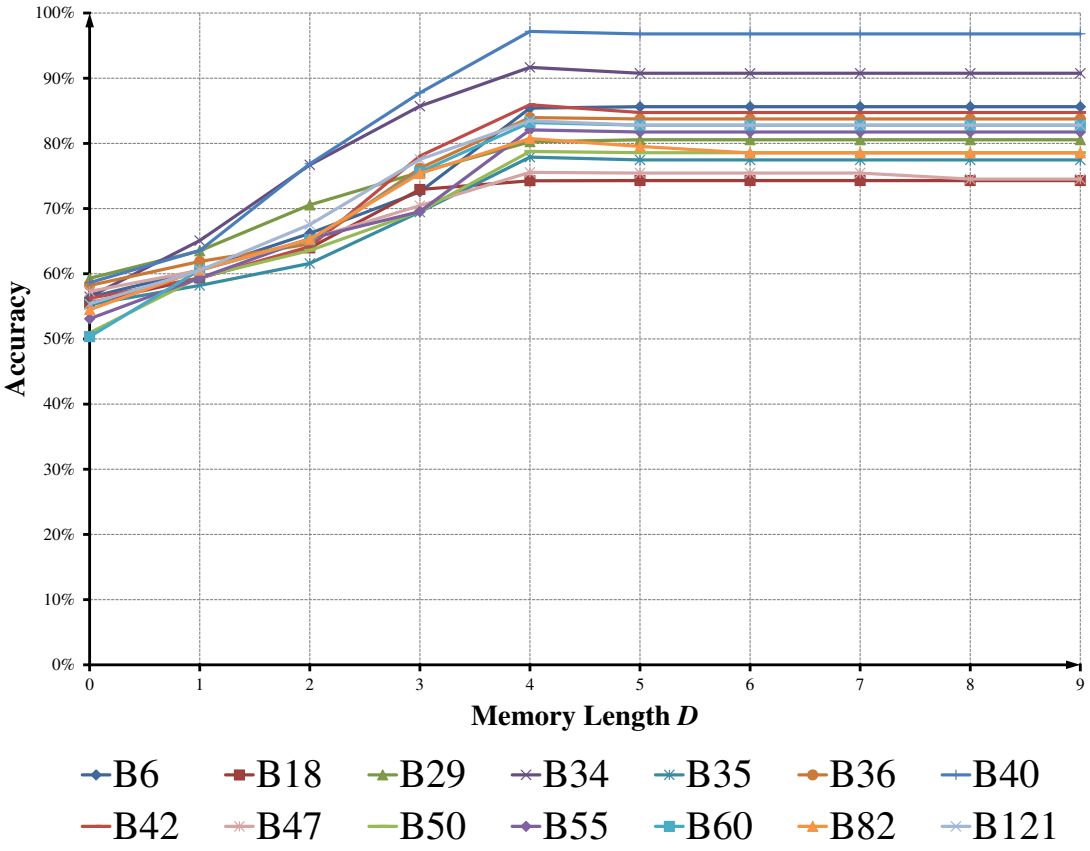
Note: B6, B18, B29, B34, B35, B36, B40, B42, B47, B50, B55, B60, B82 and B121 are the protein folds given in Table 3.3.

Figure 3.11: Average values of *accuracy* for the PPM IndelFR predictor for different protein folds selected from the *All-β protein* class for various values of the memory length D.



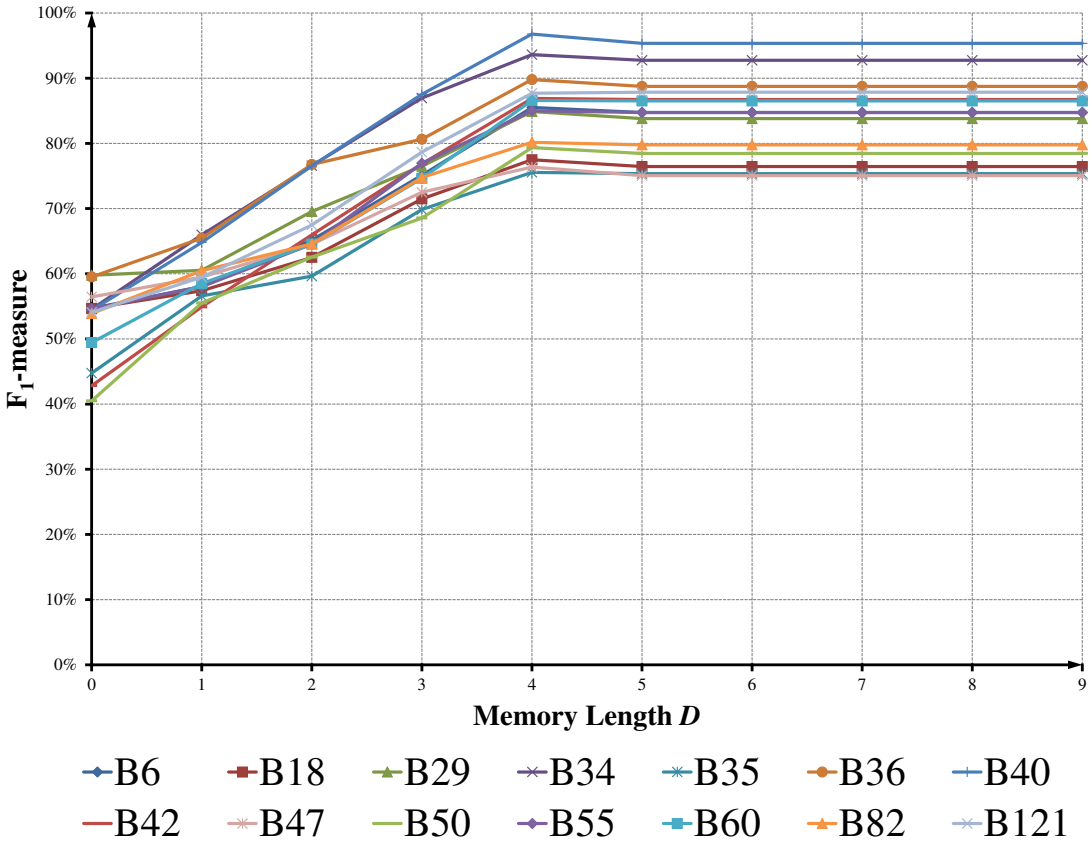
Note: B6, B18, B29, B34, B35, B36, B40, B42, B47, B50, B55, B60, B82 and B121 are the protein folds given in Table 3.3.

Figure 3.12: Average values of F_1*–measure* for the PPM IndelFR predictor for different protein folds selected from the *All-*β *protein* class for various values of the memory length D.



Note: B6, B18, B29, B34, B35, B36, B40, B42, B47, B50, B55, B60, B82 and B121 are the protein folds given in Table 3.3.

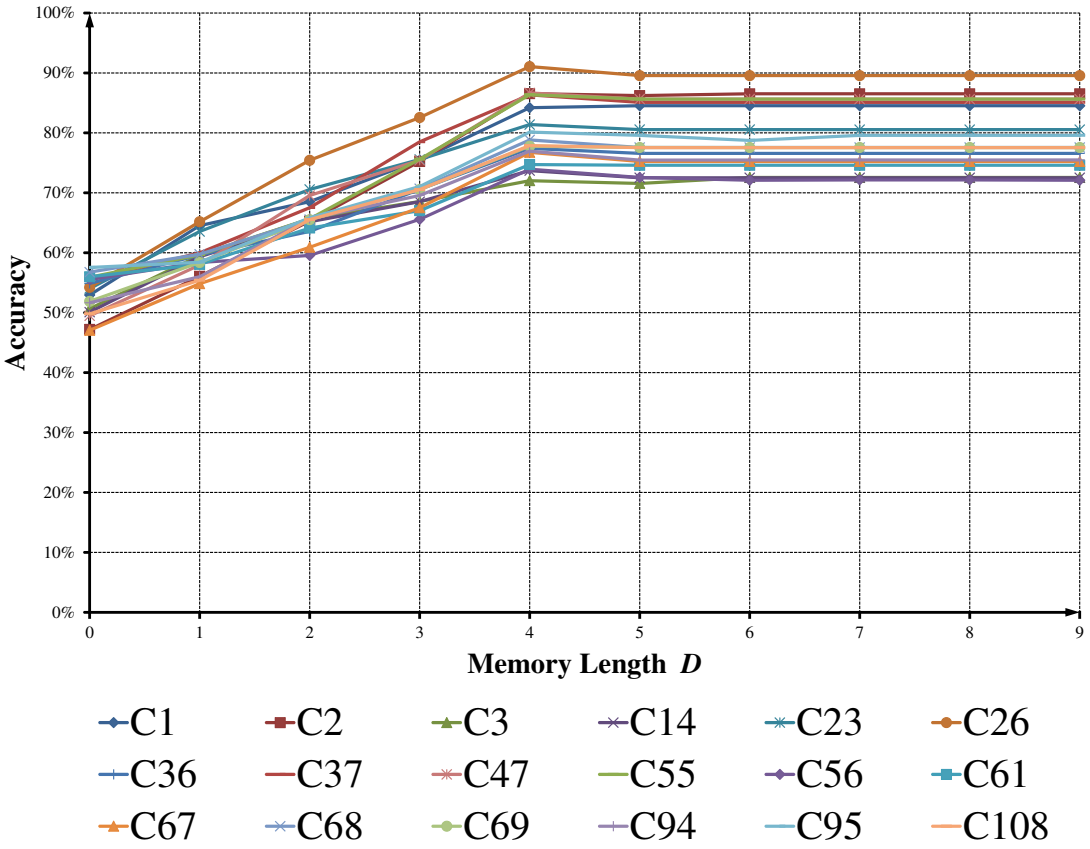
Figure 3.13: Average values of *accuracy* for the PST IndelFR predictor for different protein folds selected from the *All-β protein* class for various values of the memory length D.



Note: B6, B18, B29, B34, B35, B36, B40, B42, B47, B50, B55, B60, B82 and B121 are the protein folds given in Table 3.3.

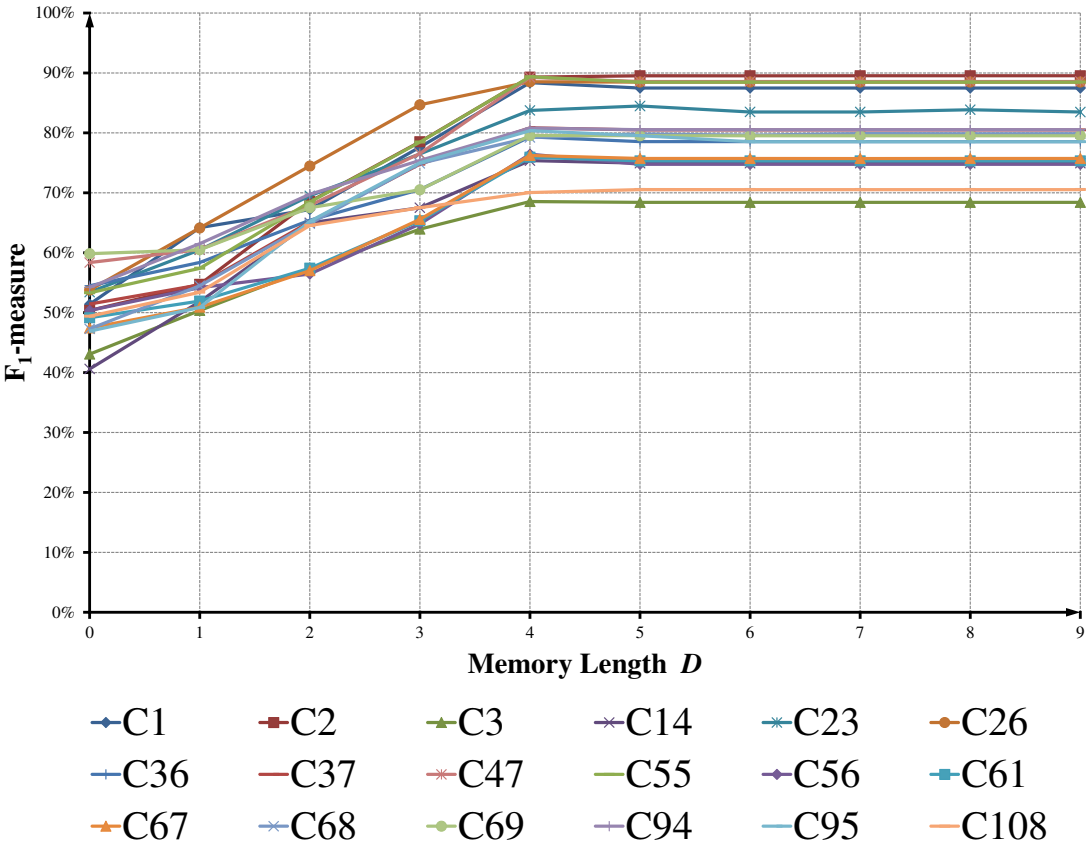
Figure 3.14: Average values of F_1–*measure* for the PST IndelFR predictor for different protein folds selected from the *All-*β *protein* class for various values of the memory length D.

The average accuracy and F_1–measure values of the PPM predictor for various values of the memory length D, $D \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, are obtained for each of the 18 chosen protein folds from the α *and* β *protein (a/b)* class, and are shown in Figures 3.15 and 3.16, respectively. It is observed from these figures that the best choice for the memory length D is 4. Further, the accuracy varies from 72% to 91% and the F_1–measure from 69% to 89% for the various folds. The average accuracy and F_1–measure values of the PPM predictor, with a memory length of 4, over the α *and* β *protein (a/b)* class are 83% and 84%, respectively. In a similar manner, the average accuracy and F_1–measure values of the PST predictor for various values of the memory length D, $D \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, are obtained for each the 18 chosen protein folds from the α *and* β *protein (a/b)* class , and are shown in Figures 3.17 and 3.18, respectively. The results for the PST predictor strongly suggest that the best choice for the memory length D is again 4. Further, the accuracy varies from 69% to 89% and the F_1–measure from 66% to 91% for the various folds. The average accuracy and F_1-measure values of the PST predictor, with a memory length of 4, over the α *and* β *protein (a/b)* class are 80% and 83%, respectively.



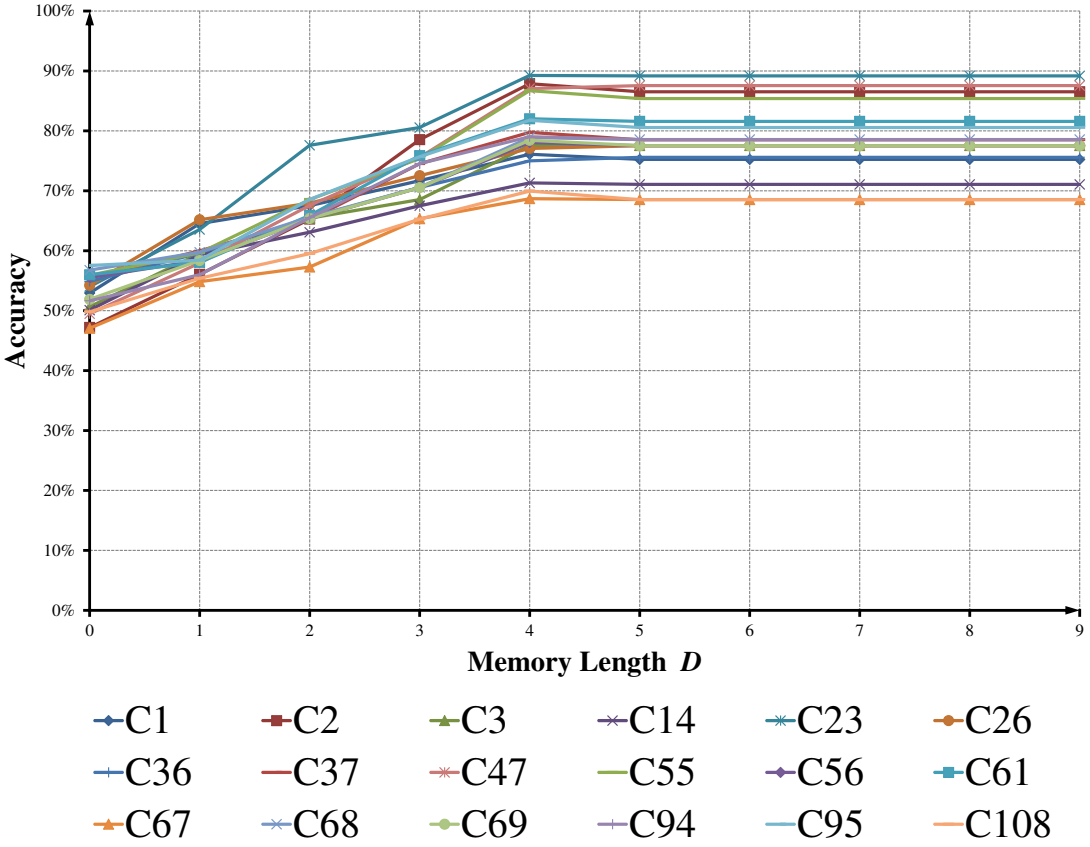
Note: C1, C2, C3, C14, C23, C26, C36, C37, C47, C55, C56, C61, C67, C68, C69, C94, C95 and C108 are the protein folds given in Table 3.4.

Figure 3.15: Average values of *accuracy* for the PPM IndelFR predictor for different protein folds selected from the α *and* β *protein (a/b)* class for various values of the memory length D.



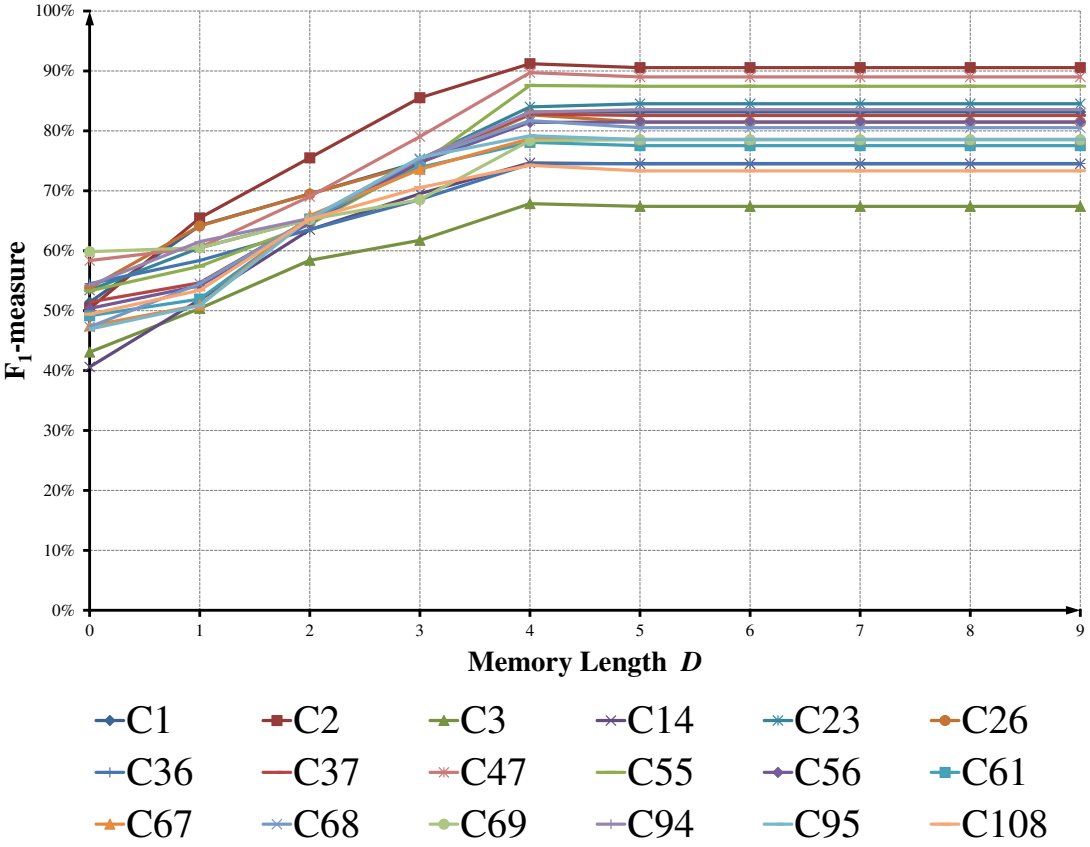
Note: C1, C2, C3, C14, C23, C26, C36, C37, C47, C55, C56, C61, C67, C68, C69, C94, C95 and C108 are the protein folds given in Table 3.4.

Figure 3.16: Average values of F_1*–measure* for the PPM IndelFR predictor for different protein folds selected from the *α and β protein (a/b)* class for various values of the memory length D.



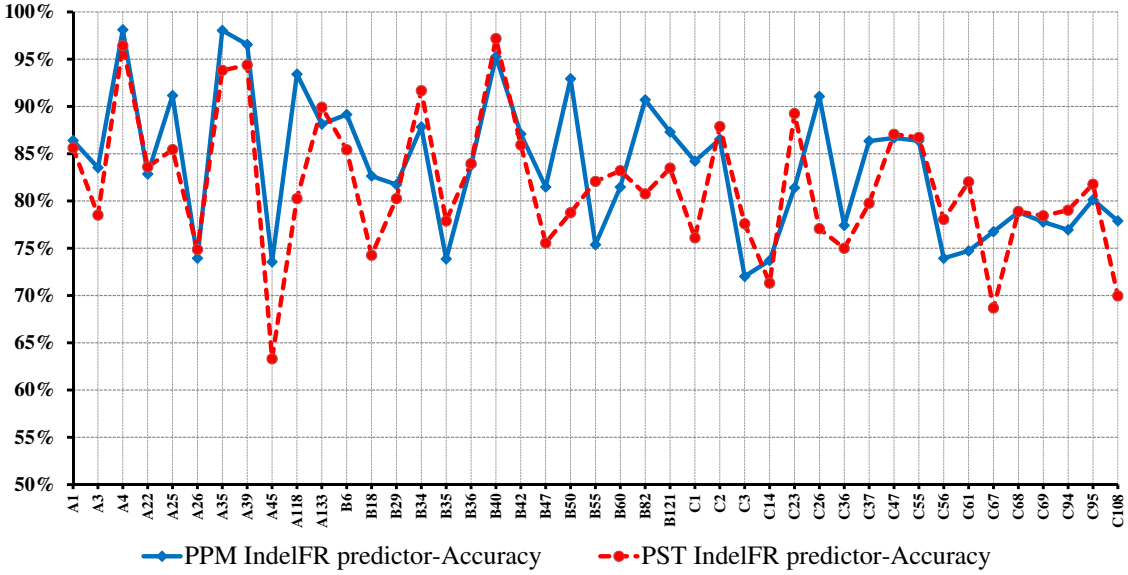
Note: C1, C2, C3, C14, C23, C26, C36, C37, C47, C55, C56, C61, C67, C68, C69, C94, C95 and C108 are the protein folds given in Table 3.4.

Figure 3.17: Average values of *accuracy* for the PST IndelFR predictor for different protein folds selected from the α *and* β *protein (a/b)* class for various values of the memory length D.



Note: C1, C2, C3, C14, C23, C26, C36, C37, C47, C55, C56, C61, C67, C68, C69, C94, C95 and C108 are the protein folds given in Table 3.4.

Figure 3.18: Average values of F_1*–measure* for the PST IndelFR predictor for different protein folds selected from the α *and* β *protein (a/b)* class for various values of the memory length D.

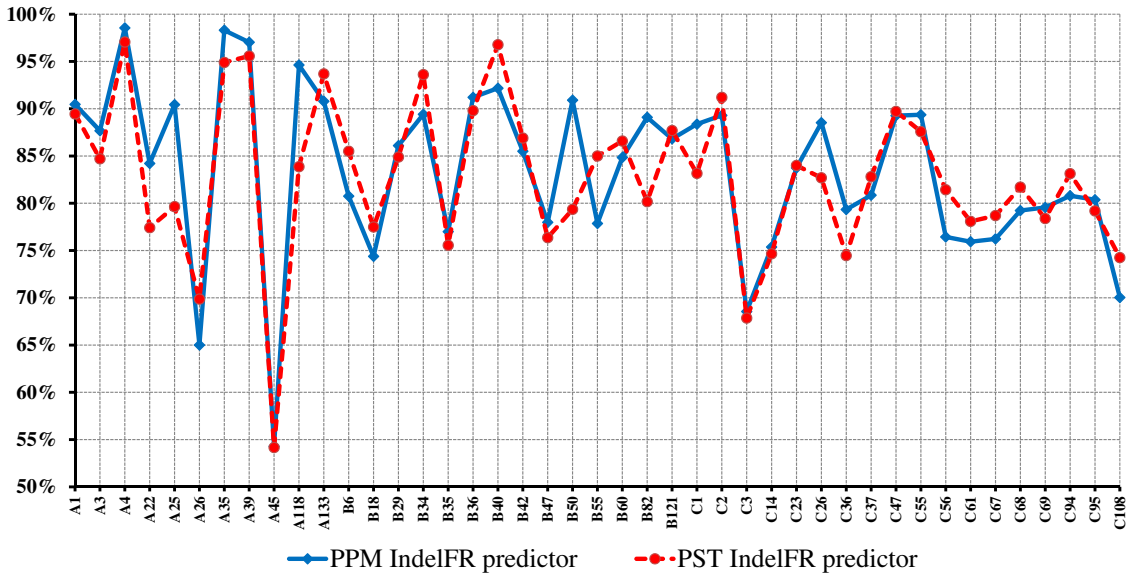


Note: A1, A3, A4, A22, A25, A26, A35, A39, A45, A118, A133, B6, B18, B29, B34, B35, B36, B40, B42, B47, B50, B55, B60, B82, B121, C1, C2, C3, C14, C23, C26, C36, C37, C47, C55, C56, C61, C67, C68, C69, C94, C95, and C108 are the protein folds given in Tables 3.2, 3.3 and 3.4.

Figure 3.19: Average values of *accuracy* for the proposed PPM and PST IndelFR predictors with $D = 4$ for different protein folds selected from the *All-α protein*, *All-β protein*, and *α and β protein (a/b)* protein classes for memory length $D = 4$.

The above results show that the best choice for the memory length D is 4 for all the selected protein folds. In addition, the results indicate that the proposed predictors perform better on those protein folds that have a large number of protein sequences. The average accuracy and F_1-measure values of the proposed PPM and PST predictors with $D = 4$, for the selected 11, 14 and 18 protein folds from the three protein classes are shown in Figures 3.19 and 3.20, respectively.

Average performances in terms of the accuracy and F_1-measure, over all the protein sequences contained in the 11, 14 and 18 protein folds of the IndelFR database belonging to the *All-α protein*, *All-β protein*, and *α and β protein (a/b)* classes, respectively, for the two proposed predictors are given in Table 3.6. It is seen from this table that the proposed PPM and PST predictors with $D = 4$ provide about the same average performance.



Note: A1, A3, A4, A22, A25, A26, A35, A39, A45, A118, A133, B6, B18, B29, B34, B35, B36, B40, B42, B47, B50, B55, B60, B82, B121, C1, C2, C3, C14, C23, C26, C36, C37, C47, C55, C56, C61, C67, C68, C69, C94, C95, and C108 are the protein folds given in Tables 3.2, 3.3 and 3.4.

Figure 3.20: Average values of F_1–*measure* for the proposed PPM and PST IndelFR predictors with $D = 4$ for different protein folds selected from the *All-*α *protein*, *All-*β *protein*, and α *and* β *protein (a/b)* protein classes for memory length $D = 4$.

Table 3.6
Average values of *accuracy* and F_1–*measure* for the proposed PPM and PST predictors with $D = 4$ and that obtained using HMMER 3.0 over all the protein sequences contained in the selected 11, 14 and 18 protein folds from the All-α protein, All-β protein, and α and β protein (a/b) classes, respectively, for (a) IndelFR database, (b) SABmark-Superfamily set and (c) SABmark-Twilight set.

	Protein classes	(a)		(b)		(c)	
		Accuracy	F_1	Accuracy	F_1	Accuracy	F_1
PPM IndelFR Predictor	All-α	91%	92%	76%	79%	74%	76%
	All-β	86%	85%	74%	80%	75%	78%
	α and β	83%	84%	77%	79%	74%	78%
PST IndelFR Predictor	All-α	88%	89%	76%	79%	71%	75%
	All-β	84%	86%	76%	83%	75%	81%
	α and β	80%	83%	76%	80%	73%	77%
HMMER alignment software	All-α	43%	49%	57%	64%	59%	66%
	All-β	43%	49%	61%	68%	61%	70%
	α and β	46%	56%	59%	69%	58%	68%

3.5.3 Prediction in SABmark 1.65

In order to have a more stringent assessment of the performance of the proposed predictors, we now test the two predictors with $D = 4$ on the sequence alignment benchmark (SABmark 1.65) [39]. It should be noted that the SABmark is generated from the SCOP database, and covers the entire known protein fold space with two sets, referred to as the *Superfamily set* and the *Twilight set*. The similarity level between any two protein sequences is less than 50% in the Superfamily set, while it is at most 25% in the Twilight set, in contrast to that in IndelFR database, which contains protein sequences that have a similarity level which could be as high as 95%.

To evaluate the performance of the proposed PPM and PST predictors on the Superfamily and Twilight sets, we select protein sequences from the protein folds belonging to these sets; the folds chosen are only those for which the predictors have already been designed using the IndelFR database. The average accuracy and F_1–

measure of the proposed predictors for each of the above protein folds are given in Tables 3.7 and 3.8, respectively. The average performance of the proposed predictors are also given in Table 3.6 for the Superfamily and Twilight sets. These results show that the proposed predictors are still able to predict the IndelFRs in the selected protein folds from both the sets with large values of accuracy and F_1-measure, even though the similarity level between any two protein sequences is at most 50% in the case of the Superfamily set, and at most 25% in the case of the Twilight set. The results show that the performances of the two proposed predictors are almost the same for both the sets, the average accuracies being around 75% and the average F_1-measures being about 79%. The average accuracy values and average F_1-measure values for the proposed IndelFR predictors for the selected protein folds from the three protein classes are shown in Figures 3.21 and 3.22 for the Superfamily and Twilight sets, respectively. It is clear from these figures as well as from Tables 3.7 and 3.8 that the values of the performance metrics, accuracy and F_1-measure, are higher than their average values for those folds in which the number of protein sequences are large (for example, for the protein folds A1, C1 and C23), and they are lower than the average values for those folds in which the number of protein sequences are smaller (for example, for the protein folds A26, B35 and C14).

Table 3.7
Average values of *accuracy* and F_1–*measure* of the proposed IndelFR predictors with memory length $D = 4$ for the Superfamily set from the SABmark benchmark for the selected protein folds from different protein classes (see Table 3.2, 3.3 and 3.4).

		PPM IndelFR predictor		**PST IndelFR predictor**	
Protein fold label	**Number of sequences**	F_1	**Accuracy**	F_1	**Accuracy**
A1	26	79%	75%	85%	80%
A3	22	90%	86%	79%	81%
A4	49	78%	75%	87%	78%
A22	10	80%	75%	80%	74%
A25	14	73%	76%	74%	80%
A26	13	53%	64%	55%	59%
A35	11	83%	83%	80%	80%
A39	22	79%	72%	71%	69%
A45	17	79%	81%	80%	79%
A118	20	81%	74%	79%	71%
A133	6	84%	72%	76%	77%
B6	15	66%	64%	75%	70%
B18	21	84%	77%	84%	78%
B29	14	82%	75%	82%	77%
B34	37	78%	65%	82%	69%
B35	1	49%	59%	50%	49%
B36	14	84%	75%	87%	78%
B40	47	84%	82%	80%	75%

(Continued on next page)

Table 3.7 (Continued from previous page)

Protein fold label	Number of sequences	PPM IndelFR predictor		PST IndelFR predictor	
		F_1	Accuracy	F_1	Accuracy
B42	22	77%	66%	89%	81%
B47	21	85%	81%	84%	77%
B50	9	75%	71%	79%	69%
B55	18	70%	73%	84%	77%
B60	19	79%	72%	84%	82%
B82	18	75%	76%	79%	75%
B121	41	82%	77%	83%	78%
C1	84	81%	77%	84%	80%
C2	16	81%	78%	75%	72%
C3	19	62%	66%	59%	65%
C14	5	59%	70%	53%	63%
C23	32	90%	87%	83%	82%
C26	18	80%	80%	82%	70%
C36	9	78%	67%	76%	62%
C37	21	82%	83%	81%	79%
C47	22	79%	77%	85%	85%
C55	25	79%	76%	80%	72%
C56	9	73%	72%	82%	76%
C61	9	74%	70%	77%	69%
C67	20	82%	81%	85%	79%
C68	4	85%	77%	84%	76%

(Continued on next page)

Table 3.7 (Continued from previous page)

Protein fold label	**Number of sequences**	**PPM IndelFR predictor**		**PST IndelFR predictor**	
		F_1	**Accuracy**	F_1	**Accuracy**
C69	19	80%	77%	84%	80%
C94	16	81%	75%	84%	83%
C95	13	64%	65%	77%	66%
C108	5	68%	64%	64%	64%

Table 3.8
Average values of *accuracy* and F_1–*measure* of the proposed IndelFR predictors with memory length $D = 4$ for the Twilight set from the SABmark benchmark for the selected protein folds from different protein classes (see Table 3.2, 3.3 and 3.4).

		PPM IndelFR predictor		**PST IndelFR predictor**	
Protein fold label	**Number of sequences**	F_1	**Accuracy**	F_1	**Accuracy**
A1	11	81%	73%	75%	80%
A3	15	72%	65%	73%	61%
A4	21	72%	73%	84%	79%
A22	8	80%	75%	73%	72%
A25	9	70%	73%	74%	72%
A26	9	54%	66%	55%	59%
A35	10	76%	82%	74%	69%
A39	12	84%	78%	72%	70%
A45	8	74%	68%	70%	68%
A118	20	87%	82%	82%	73%
A133	3	64%	69%	63%	72%
B6	12	74%	72%	80%	69%
B18	16	85%	78%	88%	84%
B29	10	81%	69%	77%	70%
B34	22	78%	82%	80%	75%
B35	3	80%	72%	77%	72%
B36	7	72%	68%	87%	78%
B40	23	84%	83%	74%	77%
B42	9	71%	65%	93%	87%

(Continued on next page)

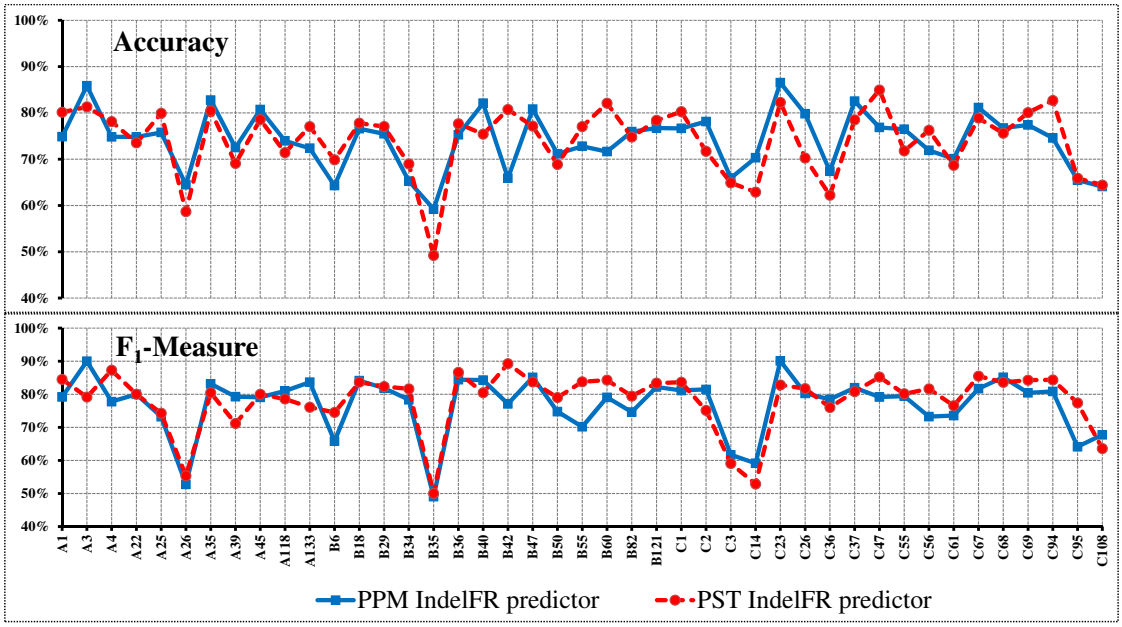
Table 3.8 (Continued from previous page)

Protein fold label	Number of sequences	PPM IndelFR predictor		PST IndelFR predictor	
		F_1	Accuracy	F_1	Accuracy
B47	8	73%	66%	78%	64%
B55	11	69%	73%	74%	61%
B60	7	79%	71%	85%	76%
B82	14	71%	74%	79%	75%
B121	23	83%	78%	81%	80%
C1	17	83%	78%	86%	80%
C2	18	74%	70%	79%	74%
C3	6	62%	66%	59%	65%
C14	3	67%	73%	46%	49%
C23	15	79%	74%	73%	69%
C26	12	80%	76%	77%	76%
C36	3	78%	67%	67%	58%
C37	21	84%	82%	82%	80%
C47	22	82%	80%	80%	75%
C55	18	79%	75%	77%	72%
C56	7	65%	64%	61%	62%
C61	5	68%	66%	59%	55%
C67	6	79%	72%	87%	81%
C68	4	83%	72%	80%	72%
C69	20	80%	77%	84%	80%
C94	9	81%	75%	79%	78%

(Continued on next page)

Table 3.8 (Continued from previous page)

		PPM IndelFR predictor		**PST IndelFR predictor**	
Protein fold label	**Number of sequences**	F_1	**Accuracy**	F_1	**Accuracy**
C95	7	53%	61%	68%	63%
C108	4	68%	55%	63%	64%

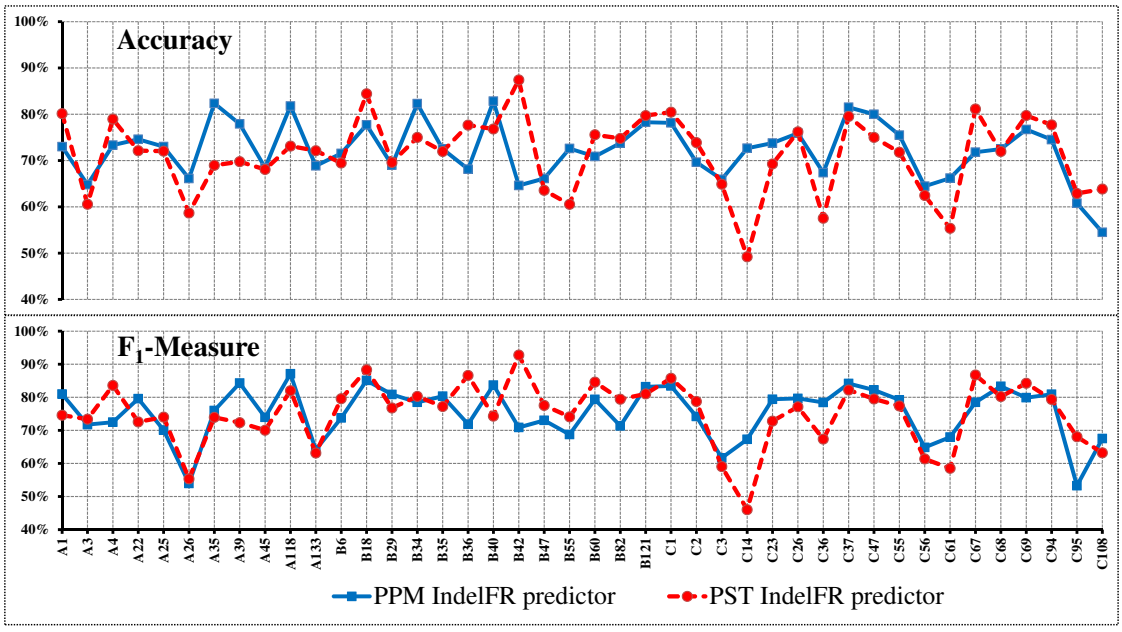


Note: A1, A3, A4, A22, A25, A26, A35, A39, A45, A118, A133, B6, B18, B29, B34, B35, B36, B40, B42, B47, B50, B55, B60, B82, B121, C1, C2, C3, C14, C23, C26, C36, C37, C47, C55, C56, C61, C67, C68, C69, C94, C95, and C108 are the protein folds given in Tables 3.2, 3.3 and 3.4.

Figure 3.21: Average values of *accuracy* and F_1*–measure* of the proposed IndelFR predictors for the Superfamily set from the SABmark benchmark for the selected protein folds from the *All-*α *protein*, *All-*β *protein*, and α *and* β *protein (a/b)* protein classes.

3.5.4 Comparison with HMMER

The performance of the proposed predictors with $D = 4$ is now compared to that obtained using the latest version of the alignment software HMMER, HMMER 3.0 [40]. HMMER 3.0 implements the alignment of a protein sequence with pHMM representing a particular protein family. A collection of pHMMs covering many protein families is available in the Pfam database [84]. In order to be able to make this comparison, it is necessary to find the Pfam pHMMs for all the protein families that belong to a protein fold for which the PPM and PST predictors have already been designed using the IndelFR database. The protein families for the selected 11, 14, and 18 protein folds from *All–*α *proteins*, *All–*β *proteins* and α *and* β *proteins*



Note: A1, A3, A4, A22, A25, A26, A35, A39, A45, A118, A133, B6, B18, B29, B34, B35, B36, B40, B42, B47, B50, B55, B60, B82, B121, C1, C2, C3, C14, C23, C26, C36, C37, C47, C55, C56, C61, C67, C68, C69, C94, C95, and C108 are the protein folds given in Tables 3.2, 3.3 and 3.4.

Figure 3.22: Average values of *accuracy* and F_1*–measure* of the proposed IndelFR predictors for the Twilight set from the SABmark benchmark for the selected protein folds from the *All-*α *protein*, *All-*β *protein*, and α *and* β *protein (a/b)* protein classes.

(a/b) are given in Appendix (see Tables A.1, A.2 and A.3).

Prediction performance using HMMER 3.0 obtained on the IndelFR database for the selected 11, 14, and 18 protein folds are shown in Tables 3.9, 3.10 and 3.11, respectively. Furthermore, prediction performance using HMMER 3.0 obtained on the Superfamily and Twilight sets for the selected 11, 14, and 18 protein folds are shown in Tables 3.12, 3.13 and 3.14, respectively. The average performances obtained using HMMER 3.0 on the IndelFR database and on the Superfamily and Twilight sets, are also included in Table 3.6. These results indicate that the proposed predictors significantly outperform that obtained using HMMER 3.0 in terms of both accuracy and F_1-measure.

It should be noted that the proposed IndelFR predictors are more general than when HMMER 3.0 is used in that the proposed PPM or PST predictor for a given protein fold is capable of predicting the indel flanking regions for any protein sequence from any protein family in that fold, whereas HMMER 3.0 has to use different pHMMs depending on the family of the protein fold to which the protein sequence belongs. For instance, we have to design only one IndelFR predictor for the *Globin-like* fold, whereas HMMER 3.0 has to use 5 different pHMMs (see Table A.1 in Appendix).

Table 3.9
Average values of *accuracy* and F_1*–measure* of HMMER alignment software for the IndelFR database for different protein folds selected from *All-α protein* (see Table A.1 in Appendix).

Protein fold label	**Pfam families**	**HMMER alignment software**	
		F_1**–measure**	**Accuracy**
A1	Globin (PF00042)	40%	32%
	Phycobilisome (PF00502)	75%	68%
	FAD_binding_2 (PF00890)	76%	64%
	Bac_globin (PF01152)	74%	62%
	Dus (PF01207)	78%	66%
A3	Cytochrom_B_C (PF00032)	65%	52%
	Cytochrom_C (PF00034)	21%	38%
A4	Homeobox (PF00046)	60%	46%
	TetR_N (PF00440)	57%	43%
	MarR (PF01047)	53%	40%
	HxlR (PF01638)	55%	42%
	TrmB (PF01978)	50%	37%
	HTH_5 (PF01022)	56%	42%
	HTH_8 (PF02954)	64%	50%
	PadR (PF03551)	60%	46%
	Myb_DNA-bind_6 (PF13921)	64%	51%
A22	Histone (PF00125)	39%	50%
	TBP (PF00352)	53%	38%
A25	Ferritin (PF00210)	30%	42%
	Ribonuc_red_sm (PF00268)	47%	42%

(Continued on next page)

Table 3.9 (Continued from previous page)

Protein fold label	**Pfam families**	**HMMER alignment software**	
		F_1–measure	**Accuracy**
	Phenol_Hydrox (PF02332)	51%	40%
A26	Hormone_1 (PF00103)	40%	33%
	IL10 (PF00726)	34%	43%
	EPO_TPO (PF00758)	39%	26%
	LIF_OSM (PF01291)	33%	34%
A35	HTH_3 (PF01381)	25%	21%
A39	efhand (PF00036)	63%	49%
A45	GST_C (PF00043)	50%	71%
	GST_N (PF02798)	34%	25%
A118	14-3-3 (PF00244)	61%	47%
	Arm (PF00514)	61%	47%
	VHS (PF00790)	59%	49%
	Sec7 (PF01369)	61%	51%
	ENTH (PF01417)	61%	48%
	TPR_2 (PF07719)	61%	47%
A133	Phospholip_A2_1 (PF00068)	12%	28%

Table 3.10
Average values of *accuracy* and F_1*–measure* of HMMER alignment software for the IndelFR database for different protein folds selected from *All-β protein* class (see Table A.2 in Appendix).

Protein fold label	**Pfam families**	**HMMER alignment software**	
		F_1**–measure**	**Accuracy**
B6	COX2 (PF00116)	48%	41%
	Copper-bind (PF00127)	33%	38%
	Cu-oxidase (PF00394)	51%	43%
B18	F5_F8_type_C (PF00754)	47%	41%
	CBM_6 (PF03422)	44%	38%
B29	Laminin_G_1 (PF00054)	72%	62%
	Lectin_legB (PF00139)	56%	47%
	Gal-bind_lectin (PF00337)	68%	58%
	Glyco_hydro_11 (PF00457)	66%	53%
	Glyco_hydro_7 (PF00840)	73%	62%
	Glyco_hydro_12 (PF01670)	72%	60%
B34	SH3_1 (PF00018)	35%	25%
	MBT (PF02820)	61%	48%
B35	Cpn10 (PF00166)	53%	40%
	ADH_zinc_N (PF00107)	64%	49%
B36	PDZ (PF00595)	14%	18%
B40	tRNA-synt_2 (PF00152)	58%	46%
	CSD (PF00313)	56%	42%
	SSB (PF00436)	58%	46%
	SNase (PF00565)	59%	46%

(Continued on next page)

Table 3.10 (Continued from previous page)

Protein fold label	**Pfam families**	**HMMER alignment software**	
		F_1–measure	**Accuracy**
	Pyrophosphatase (PF00719)	58%	47%
	tRNA_bind (PF01588)	56%	44%
	Stap_Strp_tox_C (PF02876)	59%	46%
B42	FGF (PF00167)	45%	49%
	Ricin_B_lectin (PF00652)	36%	35%
B47	Trypsin (PF00089)	17%	41%
	Peptidase_C3 (PF00548)	58%	46%
	Trypsin_2 (PF13365)	47%	42%
B50	Asp (PF00026)	25%	39%
	RVP (PF00077)	69%	60%
B55	PH (PF00169)	40%	39%
	WH1 (PF00568)	57%	48%
	PID (PF00640)	60%	51%
B60	Lipocalin (PF00061)	18%	30%
	DUF1794 (PF08768)	72%	61%
B82	cNMP_binding (PF00027)	48%	37%
	Cupin_1 (PF00190)	42%	32%
	dTDP_sugar_isom (PF00908)	49%	38%
	Cupin_2 (PF07883)	44%	33%
B121	Rhv (PF00073)	43%	36%
	Viral_coat (PF00729)	53%	44%
	Parvo_coat (PF00740)	57%	48%

(Continued on next page)

Table 3.10 (Continued from previous page)

Protein fold label	**Pfam families**	**HMMER alignment software**	
		F_1–measure	**Accuracy**
	Tymo_coat (PF00983)	57%	45%
	Peptidase_A6 (PF01829)	57%	45%
	Phage_F (PF02305)	58%	47%

Table 3.11
Average values of *accuracy* and F_1*–measure* of HMMER alignment software for the IndelFR database for different protein folds selected from α *and* β *protein (a/b)* class (see Table A.3 in Appendix).

Protein fold label	**Pfam families**	**HMMER alignment software**	
		F_1**–measure**	**Accuracy**
C1	TIM (PF00121)	59%	46%
	Alpha-amylase (PF00128)	57%	45%
	Cellulase (PF00150)	58%	45%
	Glyco_hydro_1 (PF00232)	59%	46%
	Aldo_ket_red (PF00248)	58%	46%
	Glycolytic (PF00274)	60%	47%
	Glyco_hydro_18 (PF00704)	59%	46%
	Oxidored_FMN (PF00724)	60%	47%
	FMN_dh (PF01070)	59%	47%
	MR_MLE (PF01188)	62%	48%
	AP_endonuc_2 (PF01261)	58%	46%
	RuBisCO_large_N (PF02788)	61%	47%
	Enolase_N (PF03952)	61%	47%
	Amidohydro_3 (PF07969)	60%	47%
C2	Gp_dh_N (PF00044)	63%	54%
	Ldh_1_N (PF00056)	59%	52%
	adh_short (PF00106)	54%	45%
	ADH_zinc_N (PF00107)	66%	59%
	THF_DHG_CYH (PF00763)	70%	58%
	NAD_Gly3P_dh_N (PF01210)	64%	55%

(Continued on next page)

Table 3.11 (Continued from previous page)

Protein fold label	**Pfam families**	**HMMER alignment software**	
		F_1–**measure**	**Accuracy**
	NAD_binding_2 (PF03446)	62%	54%
C3	Pyr_redox (PF00070)	41%	31%
	FAD_binding_2 (PF00890)	39%	34%
	DAO (PF01266)	38%	33%
C14	ECH (PF00378)	31%	39%
	CLP_protease (PF00574)	68%	54%
	Carboxyl_trans (PF01039)	53%	47%
C23	Response_reg (PF00072)	50%	39%
	GATase (PF00117)	54%	43%
	Flavodoxin_1 (PF00258)	55%	45%
	DJ-1_PfpI (PF01965)	58%	46%
	FMN_red (PF03358)	59%	47%
	Flavodoxin_2 (PF02525)	56%	46%
C26	tRNA-synt_1 (PF00133)	55%	41%
	ETF (PF01012)	49%	39%
	CTP_transf_2 (PF01467)	48%	41%
	NAD_synthase (PF02540)	53%	42%
C36	TPP_enzyme_M (PF00205)	65%	51%
	Transketolase_N (PF00456)	50%	43%
	E1_dh (PF00676)	61%	52%
	TPP_enzyme_N (PF02776)	44%	36%
	Transket_pyr (PF02779)	51%	49%

(Continued on next page)

Table 3.11 (Continued from previous page)

Protein fold label	**Pfam families**	**HMMER alignment software**	
		F_1–**measure**	**Accuracy**
C37	AAA (PF00004)	62%	52%
	ABC_tran (PF00005)	65%	54%
	GTP_EFTU (PF00009)	54%	46%
	Arf (PF00025)	55%	47%
	RecA (PF00154)	67%	56%
	Kinesin (PF00225)	67%	56%
	DEAD (PF00270)	64%	52%
	Helicase_C (PF00271)	69%	56%
	ADK (PF00406)	65%	53%
	Sulfotransfer_1 (PF00685)	67%	55%
	MMR_HSR1 (PF01926)	60%	50%
	AAA_2 (PF07724)	67%	55%
	AAA_5 (PF07728)	65%	54%
C47	GST_C (PF00043)	76%	64%
	Thioredoxin (PF00085)	58%	50%
	AhpC-TSA (PF00578)	57%	47%
	GST_N (PF02798)	57%	48%
	Redoxin (PF08534)	55%	47%
	Thioredoxin_2 (PF13098)	59%	50%
	GST_N_3 (PF13417)	56%	48%
	Thioredoxin_8 (PF13905)	62%	51%
C55	HSP70 (PF00012)	49%	38%

(Continued on next page)

Table 3.11 (Continued from previous page)

Protein fold label	**Pfam families**	**HMMER alignment software**	
		F_1–**measure**	**Accuracy**
	ROK (PF00480)	44%	36%
	Peptidase_M24 (PF00557)	50%	38%
	rve (PF00665)	49%	39%
	DNA_pol_B_exo2 (PF10108)	49%	37%
C56	PNP_UDP_1 (PF01048)	38%	38%
	Peptidase_M20 (PF01546)	50%	44%
	Propep_M14 (PF02244)	53%	39%
	M20_dimer (PF07687)	54%	40%
C61	Pribosyltran (PF00156)	25%	34%
C67	Aminotran_1_2 (PF00155)	39%	35%
	Cys_Met_Meta_PP (PF01053)	62%	51%
	Beta_elim_lyase (PF01212)	61%	50%
C68	Hexapep (PF00132)	58%	44%
	NTP_transferase (PF00483)	45%	34%
	CTP_transf_3 (PF02348)	47%	39%
C69	Abhydrolase_1 (PF00561)	47%	40%
	Esterase (PF00756)	56%	45%
	Abhydrolase_3 (PF07859)	54%	42%
	Abhydrolase_5 (PF12695)	41%	33%
	Abhydrolase_6 (PF12697)	35%	34%
C94	Transferrin (PF00405)	57%	47%
	LysR_substrate (PF03466)	59%	49%

(Continued on next page)

Table 3.11 (Continued from previous page)

Protein fold label	**Pfam families**	**HMMER alignment software**	
		F_1–measure	**Accuracy**
C95	Thiolase_N (PF00108)	36%	34%
	Chal_sti_synt_N (PF00195)	41%	42%
	Chal_sti_synt_C (PF02797)	62%	47%
	Thiolase_C (PF02803)	60%	44%
C108	Hydrolase (PF00702)	35%	36%
	Hydrolase_3 (PF08282)	46%	40%
	HAD (PF12710)	45%	40%
	HAD_2 (PF13419)	39%	39%

Table 3.12
Average values of *accuracy* and F_1–*measure* of HMMER alignment software for the Superfamily and the Twilight from the SABmark for the selected protein folds from the *All-α protein* class (see Table A.1 in Appendix).

Protein fold label	**Pfam families**	**HMMER alignment software**			
		Superfamily set		**Twilight set**	
		F_1	**Accuracy**	F_1	**Accuracy**
A1	Globin (PF00042)	52%	54%	65%	62%
	Phycobilisome (PF00502)	55%	52%	66%	60%
	FAD_binding_2 (PF00890)	67%	54%	70%	58%
	Bac_globin (PF01152)	65%	54%	51%	50%
	Dus (PF01207)	66%	54%	70%	60%
A3	Cytochrom_B_C (PF00032)	77%	64%	83%	73%
	Cytochrom_C (PF00034)	42%	45%	41%	37%
A4	Homeobox (PF00046)	76%	65%	79%	68%
	TetR_N (PF00440)	77%	66%	79%	67%
	MarR (PF01047)	76%	64%	77%	65%
	HxlR (PF01638)	73%	62%	72%	61%
	TrmB (PF01978)	74%	64%	73%	62%
	HTH_5 (PF01022)	72%	62%	71%	61%
	HTH_8 (PF02954)	81%	69%	78%	66%
	PadR (PF03551)	77%	66%	74%	63%
	Myb_DNA-bind_6 (PF13921)	78%	67%	78%	66%
A22	Histone (PF00125)	38%	50%	40%	51%
	TBP (PF00352)	63%	48%	64%	48%
A25	Ferritin (PF00210)	44%	51%	48%	52%
	Ribonuc_red_sm (PF00268)	54%	54%	56%	47%

(Continued on next page)

Table 3.12 (Continued from previous page)

Protein fold label	**Pfam families**	**HMMER alignment software**			
		Superfamily set		**Twilight set**	
		F_1	**Accuracy**	F_1	**Accuracy**
	Phenol_Hydrox (PF02332)	63%	52%	62%	53%
A26	Hormone_1 (PF00103)	74%	65%	78%	75%
	IL10 (PF00726)	59%	53%	68%	51%
	EPO_TPO (PF00758)	71%	55%	68%	53%
	LIF_OSM (PF01291)	73%	61%	75%	67%
A35	HTH_3 (PF01381)	60%	59%	72%	69%
A39	efhand (PF00036)	61%	50%	71%	60%
A45	GST_C (PF00043)	65%	66%	55%	57%
	GST_N (PF02798)	77%	64%	69%	56%
A118	14-3-3 (PF00244)	71%	59%	73%	60%
	Arm (PF00514)	69%	56%	73%	60%
	VHS (PF00790)	51%	42%	61%	54%
	Sec7 (PF01369)	68%	55%	73%	60%
	ENTH (PF01417)	55%	43%	64%	56%
	TPR_2 (PF07719)	71%	60%	70%	57%
A133	Phospholip_A2_1 (PF00068)	25%	39%	33%	48%

Table 3.13
Average values of *accuracy* and F_1*–measure* of HMMER alignment software for the Superfamily and the Twilight from the SABmark for the selected protein folds from the *All-β protein* class (see Table A.2 in Appendix).

Protein fold label	**Pfam families**	**HMMER alignment software**			
		Superfamily set		**Twilight set**	
		F_1	**Accuracy**	F_1	**Accuracy**
B6	COX2 (PF00116)	76%	64%	79%	68%
	Copper-bind (PF00127)	59%	52%	70%	59%
	Cu-oxidase (PF00394)	67%	59%	78%	66%
B18	F5_F8_type_C (PF00754)	69%	61%	73%	64%
	CBM_6 (PF03422)	80%	69%	80%	69%
B29	Laminin_G_1 (PF00054)	76%	66%	70%	61%
	Lectin_legB (PF00139)	72%	68%	84%	73%
	Gal-bind_lectin (PF00337)	70%	62%	71%	62%
	Glyco_hydro_11 (PF00457)	84%	73%	81%	70%
	Glyco_hydro_7 (PF00840)	86%	76%	85%	75%
	Glyco_hydro_12 (PF01670)	81%	72%	78%	71%
B34	SH3_1 (PF00018)	71%	65%	81%	71%
	MBT (PF02820)	78%	68%	75%	66%
B35	Cpn10 (PF00166)	13%	19%	18%	54%
	ADH_zinc_N (PF00107)	82%	70%	81%	71%
B36	PDZ (PF00595)	38%	47%	37%	44%
B40	tRNA-synt_2 (PF00152)	79%	68%	74%	61%
	CSD (PF00313)	80%	68%	74%	61%
	SSB (PF00436)	77%	65%	71%	57%
	SNase (PF00565)	80%	68%	75%	62%

(Continued on next page)

Table 3.13 (Continued from previous page)

Protein fold label	**Pfam families**	**HMMER alignment software**			
		Superfamily set		**Twilight set**	
		F_1	**Accuracy**	F_1	**Accuracy**
	Pyrophosphatase (PF00719)	77%	67%	71%	61%
	tRNA_bind (PF01588)	71%	60%	70%	57%
	Stap_Strp_tox_C (PF02876)	79%	68%	75%	62%
B42	FGF (PF00167)	62%	63%	93%	88%
	Ricin_B_lectin (PF00652)	60%	55%	32%	25%
B47	Trypsin (PF00089)	22%	38%	58%	55%
	Peptidase_C3 (PF00548)	79%	67%	75%	68%
	Trypsin_2 (PF13365)	67%	57%	73%	62%
B50*	Asp (PF00026)	56%	66%		
	RVP (PF00077)	41%	44%		
B55	PH (PF00169)	63%	55%	56%	48%
	WH1 (PF00568)	61%	59%	64%	61%
	PID (PF00640)	71%	66%	76%	70%
B60	Lipocalin (PF00061)	31%	40%	44%	45%
	DUF1794 (PF08768)	77%	65%	78%	65%
B82	cNMP_binding (PF00027)	65%	54%	59%	47%
	Cupin_1 (PF00190)	59%	51%	49%	42%
	dTDP_sugar_isom (PF00908)	58%	56%	62%	54%
	Cupin_2 (PF07883)	65%	54%	55%	44%
B121	Rhv (PF00073)	65%	56%	71%	61%
	Viral_coat (PF00729)	69%	58%	73%	62%
	Parvo_coat (PF00740)	75%	65%	75%	64%

(Continued on next page)

Table 3.13 (Continued from previous page)

Protein fold label	**Pfam families**	**HMMER alignment software**			
		Superfamily set		**Twilight set**	
		F_1	**Accuracy**	F_1	**Accuracy**
	Tymo_coat (PF00983)	75%	63%	73%	62%
	Peptidase_A6 (PF01829)	77%	64%	78%	65%
	Phage_F (PF02305)	76%	65%	75%	67%

**Twilight set does not have any protein sequence belong to this protein fold (B50).*

Table 3.14
Average values of *accuracy* and F_1*–measure* of HMMER alignment software for the Superfamily and the Twilight from the SABmark for the selected protein folds from the α *and* β *protein (a/b)* class (see Table A.3 in Appendix).

Protein fold label	**Pfam families**	**HMMER alignment software**			
		Superfamily set		**Twilight set**	
		F_1	**Accuracy**	F_1	**Accuracy**
C1	TIM (PF00121)	77%	64%	81%	68%
	Alpha-amylase (PF00128)	76%	65%	80%	69%
	Cellulase (PF00150)	78%	66%	79%	67%
	Glyco_hydro_1 (PF00232)	75%	66%	75%	67%
	Aldo_ket_red (PF00248)	76%	65%	81%	68%
	Glycolytic (PF00274)	78%	66%	75%	65%
	Glyco_hydro_18 (PF00704)	79%	66%	80%	67%
	Oxidored_FMN (PF00724)	78%	66%	80%	68%
	FMN_dh (PF01070)	78%	65%	80%	68%
	MR_MLE (PF01188)	78%	66%	81%	68%
	AP_endonuc_2 (PF01261)	77%	65%	78%	66%
	RuBisCO_large_N (PF02788)	79%	66%	81%	68%
	Enolase_N (PF03952)	79%	66%	81%	68%
	Amidohydro_3 (PF07969)	77%	65%	79%	67%
C2	Gp_dh_N (PF00044)	71%	57%	72%	58%
	Ldh_1_N (PF00056)	68%	56%	68%	58%
	adh_short (PF00106)	57%	55%	64%	54%
	ADH_zinc_N (PF00107)	69%	56%	66%	55%
	THF_DHG_CYH (PF00763)	72%	58%	73%	59%
	NAD_Gly3P_dh_N (PF01210)	69%	56%	67%	55%

(Continued on next page)

Table 3.14 (Continued from previous page)

Protein fold label	**Pfam families**	**HMMER alignment software**			
		Superfamily set		**Twilight set**	
		F_1	**Accuracy**	F_1	**Accuracy**
	NAD_binding_2 (PF03446)	68%	55%	64%	53%
C14	ECH (PF00378)	38%	59%	37%	47%
	CLP_protease (PF00574)	57%	42%	58%	41%
	Carboxyl_trans (PF01039)	46%	46%	34%	47%
C23	Response_reg (PF00072)	65%	58%	74%	60%
	GATase (PF00117)	63%	55%	69%	62%
	Flavodoxin_1 (PF00258)	66%	59%	63%	53%
	DJ-1_PfpI (PF01965)	75%	62%	73%	59%
	FMN_red (PF03358)	74%	63%	69%	59%
	Flavodoxin_2 (PF02525)	73%	61%	72%	58%
C26	tRNA-synt_1 (PF00133)	74%	60%	74%	60%
	ETF (PF01012)	63%	57%	61%	54%
	CTP_transf_2 (PF01467)	65%	53%	65%	52%
	NAD_synthase (PF02540)	70%	59%	67%	57%
C36	TPP_enzyme_M (PF00205)	79%	67%	79%	66%
	Transketolase_N (PF00456)	59%	61%	42%	57%
	E1_dh (PF00676)	76%	64%	72%	60%
	TPP_enzyme_N (PF02776)	34%	46%	52%	55%
	Transket_pyr (PF02779)	77%	66%	79%	67%
C37	AAA (PF00004)	61%	48%	71%	58%
	ABC_tran (PF00005)	65%	52%	71%	59%
	GTP_EFTU (PF00009)	64%	51%	65%	56%

(Continued on next page)

Table 3.14 (Continued from previous page)

Protein fold label	**Pfam families**	**HMMER alignment software**			
		Superfamily set		**Twilight set**	
		F_1	**Accuracy**	F_1	**Accuracy**
	Arf (PF00025)	67%	52%	69%	59%
	RecA (PF00154)	66%	52%	72%	59%
	Kinesin (PF00225)	67%	53%	73%	59%
	DEAD (PF00270)	64%	51%	73%	60%
	Helicase_C (PF00271)	68%	52%	74%	60%
	ADK (PF00406)	67%	52%	73%	60%
	Sulfotransfer_1 (PF00685)	63%	52%	73%	60%
	MMR_HSR1 (PF01926)	64%	51%	71%	58%
	AAA_2 (PF07724)	64%	50%	73%	60%
	AAA_5 (PF07728)	62%	50%	70%	57%
C47	GST_C (PF00043)	78%	67%	83%	73%
	Thioredoxin (PF00085)	52%	53%	66%	60%
	AhpC-TSA (PF00578)	63%	59%	73%	66%
	GST_N (PF02798)	74%	63%	62%	53%
	Redoxin (PF08534)	63%	60%	70%	65%
	Thioredoxin_2 (PF13098)	60%	57%	68%	62%
	GST_N_3 (PF13417)	74%	63%	63%	54%
	Thioredoxin_8 (PF13905)	65%	58%	74%	66%
C55	HSP70 (PF00012)	62%	52%	64%	56%
	ROK (PF00480)	67%	54%	69%	55%
	Peptidase_M24 (PF00557)	70%	56%	71%	57%
	rve (PF00665)	64%	50%	68%	53%

(Continued on next page)

Table 3.14 (Continued from previous page)

Protein fold label	**Pfam families**	**HMMER alignment software**			
		Superfamily set		**Twilight set**	
		F_1	**Accuracy**	F_1	**Accuracy**
	DNA_pol_B_exo2 (PF10108)	67%	53%	68%	54%
C56	PNP_UDP_1 (PF01048)	77%	63%	68%	52%
	Peptidase_M20 (PF01546)	78%	65%	70%	54%
	Propep_M14 (PF02244)	78%	64%	68%	52%
	M20_dimer (PF07687)	60%	57%	49%	53%
C61	Pribosyltran (PF00156)	54%	51%	53%	52%
C67	Aminotran_1_2 (PF00155)	52%	54%	45%	47%
	Cys_Met_Meta_PP (PF01053)	69%	64%	79%	67%
	Beta_elim_lyase (PF01212)	74%	64%	76%	65%
C68	Hexapep (PF00132)	85%	75%	84%	74%
	NTP_transferase (PF00483)	67%	60%	65%	58%
	CTP_transf_3 (PF02348)	69%	63%	69%	64%
C69	Abhydrolase_1 (PF00561)	69%	60%	68%	60%
	Esterase (PF00756)	74%	62%	71%	60%
	Abhydrolase_3 (PF07859)	70%	59%	71%	59%
	Abhydrolase_5 (PF12695)	64%	53%	54%	45%
	Abhydrolase_6 (PF12697)	60%	54%	54%	50%
C94	Transferrin (PF00405)	76%	63%	75%	60%
	LysR_substrate (PF03466)	66%	58%	75%	61%
C95	Thiolase_N (PF00108)	45%	49%	38%	50%
	Chal_sti_synt_N (PF00195)	51%	43%	47%	38%
	Chal_sti_synt_C (PF02797)	67%	51%	62%	46%

(Continued on next page)

Table 3.14 (Continued from previous page)

Protein fold label	**Pfam families**	**HMMER alignment software**			
		Superfamily set		**Twilight set**	
		F_1	**Accuracy**	F_1	**Accuracy**
	Thiolase_C (PF02803)	65%	51%	59%	45%
C108	Hydrolase (PF00702)	49%	48%	52%	48%
	Hydrolase_3 (PF08282)	56%	53%	52%	54%
	HAD (PF12710)	56%	46%	56%	48%
	HAD_2 (PF13419)	51%	49%	56%	49%

3.6 Summary

In this chapter, a variable–order Markov model–based scheme to predict indel flanking regions (IndelFRs) in a protein sequence for a given protein fold has been proposed [36]. In this scheme, two predictors, referred to as the PPM IndelFR and PST IndelFR predictors, have been designed based on *prediction by partial match* (PPM) [37] and *probabilistic suffix tree* (PST) [38], respectively. The performance of the proposed PPM and PST IndelFR predictors have been evaluated. For this purpose, 43 protein folds from different protein classes have been selected. The k-fold cross-validation method has been used for training and testing the proposed IndelFR predictors, where $k = 10$. In the training phase, the IndelFR predictor for a given protein fold has been trained using the indel flanking regions listed in the IndelFR database. In the testing phase, the trained predictors have been tested first on the protein sequences from the same protein fold belonging to the IndelFR database and then, on the set of protein sequences from the same protein fold but belonging to the sequence alignment benchmark (SABmark 1.65) [39]. Finally, the performance of the two proposed predictors has been compared to that using the latest version of the alignment software HMMER, HMMER 3.0 [40].

It has been shown through extensive performance evaluation that the best choice for the memory length D is 4 for all the selected protein folds. This indicates that increasing the length of the context beyond 4 does not seem to improve the performance of the proposed predictors. This is because as the length of the context becomes larger, the probability of appearance of any amino acid given this long context becomes very small. Thus, the escape mechanism may be triggered a number of times and the overall probability of the amino acids in the window would become small. The results have shown that the proposed predictors are able to predict the IndelFRs in the selected protein folds with large values for accuracy and F_1-measure. The results have also shown that if one is interested only in predicting

the indel flanking regions in protein sequences, then it would be preferable to use the proposed predictors instead of using HMMER 3.0 in view of the substantially superior performance of the former. It should be noted that if HMMER 3.0 is used for prediction, one would need as many pHMMs as the number of families in a given fold, while only one proposed predictor is needed for a given fold. It should be noted that the proposed IndelFR predictors have been built in a fully automated manner without utilizing any prior assumption about the occurrence of mutations in the protein sequences, as in the case of scoring schemes.

In the next chapter, the information on the predicted location of indel flanking regions (IndelFRs) will be employed in introducing a new *variable gap penalty* (VGP) function for the alignment of multiple protein sequences.

Chapter 4

MSAIndelFR: Multiple Protein Sequence Alignment

4.1 Introduction

In the preceding chapter, the two IndelFR predictors have been proposed. The performance evaluation of these predictors have shown that they are able to predict indel flanking regions (IndelFRs) with large values for *accuracy* and F_1-*measure*. The proposed IndelFR predictors compute the left and right average log–loss values for each position in the protein sequence, and then uses Algorithm 1 in Section 3.4 to identify the predicted locations of IndelFRs in the protein sequence.

In this chapter, we propose a novel and efficient algorithm [85] for multiple sequence alignment using the information on the predicted locations of IndelFRs and the computed average log-loss values provided by the PPM IndelFR predictor designed in Chapter 3. The key innovation of this algorithm, refer to as MSAIndelFR algorithm, is the use of this information in proposing a new *variable gap penalty* (VGP) function, wherein the gap opening penalty is position–specific and the gap extension penalty is region–specific. In [3, 5, 17–19], it has been reported that there exists a strong relationship between indel mutations and their flanking regions. Therefore, by integrating the information provided by the IndelFR predictors

to the gap penalty function, a more accurate alignment can be expected.

This chapter starts by giving in Section 4.2 a brief overview of the PPM IndelFR predictor proposed in the preceding chapter. In Section 4.3, the proposed algorithm for multiple sequence alignment is developed. First, a variable gap penalty function and FASTA format are introduced. Then, the alignment strategy and the generation of alignment using dynamic programming with variable gap penalty are discussed. Experimental results on the performance of the proposed algorithm as well as those on the six most–widely used alignment algorithms are presented and compared in Section 4.4. Finally, in Section 4.5, a summary of the chapter along with some concluding remarks on the proposed MSAIndelFR algorithm is provided.

4.2 PPM IndelFR Predictor

In the preceding chapter, a technique for building the IndelFR predictor for a given protein fold, based on the *prediction by partial match* (PPM) [37], was proposed. This PPM IndelFR predictor for a given protein fold contains two variable–order Markov models, one for predicting the left flanking and the other for predicting the right flanking regions. The performance evaluation results of the PPM IndelFR predictor have shown that the best choice for the memory length D is 4.

Given a test protein sequence $\mathbf{S}^n = s_1 s_2 s_3 \cdots s_n$ of length n, the PPM IndelFR predictor scans it by moving a window of length $L = 10$, one amino acid at a time, to determine whether the string of amino acids within a window contains an IndelFR or not. It has been noted that the impact of an indel on its flanking regions reduces dramatically as we move away from the indel, and that it is negligible after 10 amino acids [3].

The PPM IndelFR predictor, with $D = 4$, computes the left and right average log–loss values for each position in the protein sequence, and then uses Algorithm 1 of Section 3.4 to extract the predicted locations of IndelFRs in the protein sequence.

As in Section 3.4, the average log–loss value for window of length $L = 10$ at position i, $\mathbf{win}_i = s_i s_{i+1} \cdots s_{i+9}$, in the sequence is defined as

$$\begin{aligned} & logloss P(\mathbf{win}_i) = \\ & \quad -\frac{1}{L}\Bigg(\log P_0(s_i) + \log P_1(s_{i+1}|s_i) + \\ & \quad \log P_2(s_{i+2}|s_1 s_{i+1}) + \cdots + \\ & \quad \log P_D(s_{i+L-1}|s_{i+L-1-D} \cdots s_{i+L-2}) \Bigg) \end{aligned} \tag{4.1}$$

where the logarithm is taken to base 2. For the purpose of illustration, the left and right average log–loss values for the protein sequence *d1liab_* at different positions are shown in Figure 4.1(a) and the corresponding predicted locations of IndelFRs for *d1liab_* are shown in Figure 4.1(b).

The PPM IndelFR predictors for 11, 14 and 18 protein folds from different protein classes, *All–α proteins*, *All–β proteins* and *α and β proteins (a/b)*, respectively, have been constructed according to the technique that we developed in Subsection 3.3.1. Hence, we have 43 different PPM IndelFR predictors. It should be noted that the PPM IndelFR predictors were trained using the IndelFR database [17], which in turn provided IndelFRs for some selected protein sequences belonging to certain selected protein folds from the SCOP database [42]. In Chapter 3, it has been demonstrated that once the PPM IndelFR predictor is built for a given protein fold, it can be used to compute the average log–loss values for any protein sequence belonging to this protein fold. Hence, we will be able to compute the average log–loss values, and then use Algorithm 1 of Section 3.4 to predict the IndelFRs for protein sequences that are available in the selected protein folds, even though the IndelFR database did not provide IndelFRs for these protein sequences.

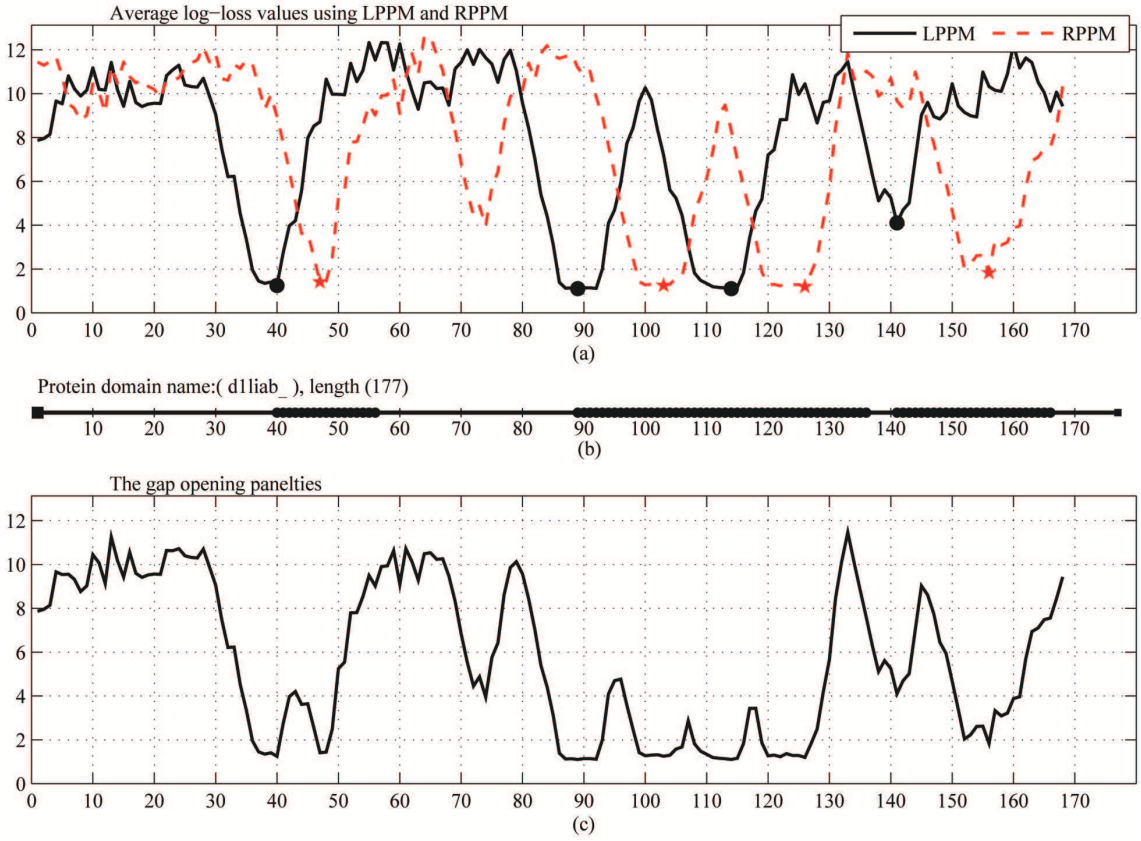


Figure 4.1: (a) Average log-loss values for the *d1liab_* using PPM IndelFR predictor. (b) Predicted locations of IndelFRs using PPM IndelFR predictor. (c) Gap opening penalties. In (a), the solid dots represent the start locations of the predicted left flanking regions and the stars that of the predicted right flanking regions.

4.3 Proposed Algorithm

In this section, we propose an algorithm for MSA, referred to as MSAIndelFR algorithm, that makes use of the computed average log–loss values and the predicted IndelFRs from the PPM IndelFR predictor. The results of the PPM IndelFR predictor proposed in Section 3.3.1 have shown that the computed average log–loss values in and around an IndelFR are much smaller than that in other regions. In view of this observation, we combine the left and right average log–loss values for any given protein sequence $\mathbf{S}^n = s_1 s_2 s_3 \cdots s_n$ of length n to propose a *position–specific gap*

opening penalty function. The proposed position–specific gap opening penalty at a particular position i in the sequence is given by

$$GPO_i = \begin{cases} \min(LPPM_i, RPPM_i), & 1 \leq i \leq (n - L + 1) \\ GPO_{(n-L+1)}, & (n - L + 1) < i \leq n \end{cases} \tag{4.2}$$

where $LPPM_i$ and $RPPM_i$ are, respectively, the left and right average log–loss values at position i. It is seen from this equation that GPO_i, for $(n-L+1) < i \leq n$, is chosen to be equal to the gap opening penalty at position $i = n - L + 1$. The gap opening penalties at different positions for *d1liab_* are shown in Figure 4.1(c).

In addition to using the gap opening penalty function GPO_i, we use the predicted IndelFRs to propose a *region–specific gap extension penalty* function. As mentioned in Section 4.1, the predicted IndelFRs are the most likely regions for the gaps to be introduced in the protein sequence, since they are strongly related to indel mutations [3, 5, 17–19]. Moreover, a single indel mutation event often affects several adjacent amino acids in a protein sequence [1]. This fact is taken into consideration in the proposed definition of the gap extension penalty at position i in the protein sequence, GPE_i :

$$GPE_i = \begin{cases} 0, & \text{if position } i \in IndelFRs \\ GPO_i, & \text{otherwise} \end{cases} \tag{4.3}$$

In other words, a zero value is assigned to GPE_i, if the gap introduced at position i is in an IndelFR, whereas it is equal to GPO_i if i is not in an IndelFR.

4.3.1 A new FASTA format

We modify the standard FASTA format to include into it the information about the position–specific gap opening penalty and the predicted locations of IndelFRs.

Hence, the input protein sequences to the proposed MSAIndelFR algorithm should be written using the modified version of FASTA format, where the position–specific gap opening penalty and the predicted locations of IndelFRs are added after the main list of amino acids of the protein sequence.

The modified version of FASTA format is the same as the standard FASTA format except that the position–specific gap opening penalties and the predicted IndelFRs are included after the main list of amino acids of a given protein sequence. The position–specific gap opening penalties are enclosed by curly brackets {} and indicated by the label "GPO". The predicted IndelFRs are included in three separate lines and enclosed by curly brackets {} : one for the start location of the left flanking regions, one for the start location of the right flanking regions, and one for the complete IndelFRs. The start locations of the left and right flanking regions are indicated by LLOC and RLOC, respectively, and the complete IndelFRs by IndelFRs. In the following example, the protein sequences, *1hjd_A* and *1pht_* of lengths 101 and 83, respectively, are written using the modified version of FASTA.

Example 5.

Sequence $1hjd\_A$

ADRKLCADQECSHPISMAVALQDYMAPDCRFLTIHRGQVVYVFSKLKGR

GRLFWGGSVQGDYYGDLAARLGYFPSSIVREDQTLKPGKVD

VKTDKWDFYCQ

{GPO: 8.555 8.788 8.345 7.982 7.448 8.230 8.003 8.146 7.328 7.101 7.772 7.211

7.486 6.681 5.500 4.650 3.804 3.095 2.076 3.393 3.481 4.052 4.703 5.419 4.775

3.749 2.838 1.802 2.633 3.924 4.803 5.259 5.987 5.221 4.348 3.465 2.354 1.475

3.059 4.008 4.953 4.942 5.388 7.387 6.451 5.501 4.468 4.721 4.445 3.684 3.545

4.444 5.085 4.599 4.323 3.972 3.883 5.886 6.325 5.669 4.973 5.349 5.418 4.978

4.826 4.952 5.250 5.545 6.498 6.273 6.104 6.729 7.094 7.917 7.465 7.379 7.841

6.890 6.547 6.711 7.057 7.263 7.778 7.959 8.226 8.271 8.558 8.752 8.449 8.714

8.476 8.757 8.757 8.757 8.757 8.757 8.757 8.757 8.757 8.757 8.757}

{LLOC: 19 38 57}

{RLOC: 28 51 65}

{IndelFR: 19 76}

Sequence 1$pht_$

AEGYQYRALYDYKKEREEDIDLHLGDILTVNKGSLVALGFSDGQEARPEEI

GWLNGYNETTGERGDFPGTYVEYIGRKKISPP

{GPO: 6.694 6.724 6.247 6.688 6.634 6.607 6.326 6.708 6.859 7.013 7.262 7.528

7.105 7.755 8.015 7.748 7.462 7.023 6.331 5.660 6.188 5.987 6.116 5.651 5.764

6.484 6.775 6.017 6.314 6.433 6.293 7.209 7.436 7.247 7.418 6.644 7.878 7.583

8.145 7.605 6.913 6.096 6.604 6.463 6.025 6.181 6.071 6.290 5.767 6.715 7.267

7.266 7.030 6.751 6.783 7.131 6.992 6.941 6.842 6.813 7.020 7.320 7.783 7.675

7.426 7.058 6.652 6.652 6.796 6.707 7.131 7.176 7.605 7.346 7.346 7.346 7.346

7.346 7.346 7.346 7.346 7.346 7.346 }

{LLOC: 24 60}

{RLOC: 31 67}

{IndelFRs: 24-41, 60-77}

■

4.3.2 Alignment strategy

The alignment strategy is based on the standard progressive alignment method for aligning multiple protein sequences [60]. First, pairwise distances between input sequences are calculated to form a distance matrix. An accurate calculation of pairwise distances can be accomplished by performing all the pairwise alignments amongst the input sequences; however, this is not practical in view of time complexity, especially when the number of sequences is large, since any pairwise alignment requires quadratic time for completion [20]. Therefore, some of the existing MSA algorithms have used the k–tuple method [62] to calculate the pairwise distances approximately. It has been shown in [11] that the Muth–Manber string matching algorithm proposed in [68] to calculate the pairwise distances is more accurate than the k–tuple method; this algorithm finds the distance between two sequences by matching patterns that contain at most one error. For example, consider two sequences $ABCABCABC$ and $ABDABDABD$ that are 67% identical. The k–tuple method (with a pattern length of 3) reports that these two sequences are not identical (i.e., share no exact patterns), while the Muth–Manber algorithm reports that these two sequences are 67% identical. In view of this, we employ the Muth–Manber algorithm to calculate the pairwise distances between the input protein sequences.

Since protein sequences are normally searched with short length patterns [11, 15, 61, 74], we search with patterns of length 3 of amino acids to calculate the pairwise distances. Then, a guide tree is constructed from the distance matrix using the unweighted pair group method with arithmetic mean (UPGMA) [65], which is the most popular method for guide tree construction and used in many MSA algorithms as the default option. Finally, sequences or profiles are aligned according to the order prescribed by the guide tree. Hence, at each internal node of the guide tree, two sequences, or two profiles or one sequence and one profile are aligned. The process of aligning sequences/profiles continues until the highest level of the guide tree is

reached. For this purpose, we use the *dynamic programming* (DP) approach along with the proposed gap penalty functions, namely, the position–specific gap opening penalty function and the region-specific gap extension penalty function, to align sequences/profiles.

4.3.3 Dynamic programming with variable gap penalty function

We assume that the input protein sequences are evolutionary related over their entire lengths. Therefore, a global alignment of the input sequences will be obtained using the DP approach. The optimal alignment in the DP approach is the alignment which has the highest score, where the score of an alignment is found by using a gap penalty function and the substitution matrix S. It should be noted that any alignment between protein sequences is intended to reflect the cost of mutational events needed to transform one sequence to the other [1, 20]. For this purpose, we use a VGP function, which has two subfunctions: the position–specific gap opening penalty function GPO_i and the region-specific gap extension penalty function GPE_i.

Let $\mathbf{A}^n = a_1 a_2 a_3 \cdots a_n$ and $\mathbf{B}^m = b_1 b_2 b_3 \cdots b_m$ be two sequences of lengths n and m, respectively. The DP approach finds the optimal alignment between $\mathbf{A}$ and $\mathbf{B}$ by computing the optimal alignments between all prefixes of $\mathbf{A}$ and $\mathbf{B}$. The amino acids in $\mathbf{A}$ and $\mathbf{B}$ are assigned to one of three possible states: aligned, gap in sequence A, or gap in sequence B during the alignment process. These states are represented by three matrices in the DP approach. Let $\mathbf{A}[1:i] = a_1 a_2 \cdots a_i$ be a prefix of sequence $\mathbf{A}$, $\mathbf{B}[1:j] = b_1 b_2 \cdots b_j$ be a prefix of sequence $\mathbf{B}$, $M(i,j)$ be the optimal score for aligning $\mathbf{A}[1:i]$ and $\mathbf{B}[1:j]$ given that a_i is aligned to b_j, $I_A(i,j)$ be the optimal score given that a_i is aligned to a gap, and $I_B(i,j)$ be the optimal score given that b_j is aligned to a gap, where $1 \leq i \leq n$ and $1 \leq j \leq m$. The recursive equations to find the various elements in the state matrices $M(i,j)$, $I_A(i,j)$, and $I_B(i,j)$ are

given by

$$M(i,j) = \max \begin{cases} M(i-1,j-1) + s(a_i, b_j), & \text{With } a_{i-1} \text{ aligned to } b_{j-1}, \text{ align } a_i \text{ to } b_j \\ I_A(i,j) + s(a_i, b_j), & \text{End a gap in } \mathbf{A}, \text{ align } a_i \text{ to } b_j \\ I_B(i,j) + s(a_i, b_j), & \text{End a gap in } \mathbf{B}, \text{ align } a_i \text{ to } b_j \end{cases} \tag{4.4}$$

$$I_A(i,j) = \max \begin{cases} M(i-1,j) - (GPO_i^A + GPE_i^A), & \text{Open a new gap in } \mathbf{A} \\ I_A(i-1,j) - GPE_i^A, & \text{Extend an old gap in } \mathbf{A} \end{cases} \tag{4.5}$$

$$I_B(i,j) = \max \begin{cases} M(i,j-1) - (GPO_j^B + GPE_j^B), & \text{Open a new gap in } \mathbf{B} \\ I_B(i,j-1) - GPE_j^B, & \text{Extend an old gap in } \mathbf{B} \end{cases} \tag{4.6}$$

with

$$\begin{aligned} &M(0,0) = 0,\ M(0,j) = GPO_1^B + \sum_{j=1}^{m} GPE_j^B, \\ &M(i,0) = GPO_1^A + \sum_{i=1}^{n} GPE_i^A \\ &I_A(0,j) = -\infty,\ I_B(i,0) = -\infty \end{aligned} \tag{4.7}$$

where $s(a_i, b_j)$ can be obtained directly from the substitution matrix S, GPO_i^A and GPE_i^A are, respectively, the gap opening and extension penalty functions for the sequence $\mathbf{A}$, and GPO_j^B and GPE_j^B are the corresponding penalty functions for the sequence $\mathbf{B}$. Once the computation of M is completed, **it** contains the maximum

alignment score, and a trace back procedure is used to retrieve the alignment between **A** and **B**.

In this section, we implement the memory efficient DP algorithm proposed in [64], which can align two sequences of lengths, say n and m ($n \geq m$), with a time complexity of $O(mn)$ and a space complexity of $O(n)$. Since it has been shown in [24] that the selection of a particular substitution matrix does not noticeably affect the alignment accuracy, and that there is little difference in the alignment accuracy using BLOSUM [21], PAM [23] or GONNET [22] as the substitution matrix, we use GONNET250 as the substitution matrix.

In order to continue aligning sequences/profiles until the highest level of the guide tree is reached, we need the gap penalty functions, GPO_i and GPE_i, for each profile. For example, consider the alignment of two sequences, say, **A** and **B**, at the lowest level of the tree to produce profile **C**. The position–specific gap opening penalty function for profile **C** is defined to be

$$GPO_i^C = \begin{cases} GPO_j^A + GPO_k^B, & \text{if } a_j \text{ is aligned with } b_k \text{ at position } i \\ GPO_j^A, & \text{if there is a gap in } \mathbf{B} \text{ at position } i \\ GPO_k^B, & \text{if there is a gap in } \mathbf{A} \text{ at position } i \end{cases} \tag{4.8}$$

where GPO_j^A , GPO_k^B and GPO_i^C are the gap opening penalty functions at positions j, k, and i for **A**, **B** and **C**, respectively. In a similar manner, we define the gap extension penalty function for **C**. This makes a gap more likely to occur at a position where a gap already exists. If there is no gap at a position i in **C**, then the gap opening penalty is increased by adding **to it** both GPO_j^A and GPO_k^B to avoid introducing gaps at the aligned positions.

As already mentioned, the internal nodes of the guide tree are visited in a bottom–up order, and for each visited node a pairwise alignment of sequences/pro-

files is computed using the DP approach along with the proposed VGP function. The MSA associated with the root node is the final alignment.

4.4 Results and Discussion

The performance of MSA algorithms are usually evaluated on alignment benchmarks containing reference alignments. In our experiments, we use four popular benchmarks, namely, BAliBASE 3.0 [32], OXBENCH [41], PREFAB 4.0 [15] and SABmark 1.65 [39] to evaluate the performance of the proposed MSAIndelFR algorithm as well as that of the six top–performing MSA algorithms, namely, Clustal W2 version 2.1, Clustal Omega version 1.2.0, MSAProbs version 0.9.7, Kalign2 version 2.04, MAFFT version 7.184 and MUSCLE version 3.8.31. For MAFFT, *auto* option is used with the maximum iterative refinement (*maxiterate* option) set to 1000, whereas the default options are used for all the other algorithms, including the proposed MSAIndelFR.

We select the reference alignments from the above four benchmarks that have protein sequences belonging to one of the 43 protein folds (see Tables A.1– A.3 in Appendix). In the preceding chapter, it has been shown that once the PPM IndelFR predictor is built for a given protein fold, it can be used to compute the average log–loss values for any protein sequence that belongs to this protein fold. Hence, we are able to compute the average log–loss values, and then use Algorithm 1 of Section 3.4 to predict the IndelFRs for protein sequences that are available in the alignment benchmarks, even though the IndelFR database does not contain IndelFRs for these protein sequences. We would like to emphasize that no training is needed in the proposed MSAIndelFR algorithm. Further, it does not make use of the protein secondary information (alpha, beta or coil) as input. It makes use of the computed average log–loss values and the predicted IndelFRs from the PPM IndelFR predictor proposed in Section 3.3. It should be noted that the PPM IndelFR

predictor does not use any of the above–mentioned four benchmarks for its training.

We use the measures, *sum-of-pairs* (SP) and *total columns* (TC) [34], which are the most commonly used metrics, to evaluate and compare the performance of the various MSA algorithms. The SP value is defined as the number of correctly aligned amino acid pairs found in the test alignment divided by the total number of aligned amino acid pairs in the *core blocks* of the reference alignment, where the core blocks of the reference alignment refer to the regions for which reliable alignments are known to exist. We use bench database (Edgar, R.C., *http://www.drive5.com/bench*) to determine the core blocks in the selected benchmarks. It should be noted that the *quality* (Q) metric used in [15] is equivalent to SP. The TC value is defined as the number of correctly aligned columns found in the test alignment divided by the total number of aligned columns in the core blocks of the reference alignment, and hence, gives the proportion of the total alignment columns that is recovered in the test alignment. A value of 1.0 for TC indicates perfect agreement between the test and reference alignments. It should be noted that the TC value is equivalent to the SP value in the case of pairwise alignment (as in the PREFAB benchmark). The SP and TC values are calculated employing the QSCORE software available at *http://www.drive5.com/qscore/*. Example 6 is given below illustrates how the SP and TC values are calculated.

Example 6.

In this example, we explain how the *sum–of–pairs* (SP) and the *total column* (TC) values are computed.

Given the reference alignment:

```
Seq [0]:  GKGDRKK
Seq [1]:  MQ-DRVK
Seq [2]:  MKKLKKH
Seq [3]:  MHIK-PL
```

and the test alignment:

```
Seq [0]:  GK-GDRKK
Seq [1]:  MQ-DRVK-
Seq [2]:  MKKLKKH-
Seq [3]:  MHIK-PL-
```

the *sum–of–pairs* (SP) value is given by

$$SP = \frac{The\ number\ of\ correclty\ aligned\ amino\ acid\ pairs\ found\ in\ the\ test\ alignment}{The\ total\ number\ of\ aligned\ amino\ acid\ pairs\ in\ the\ reference\ alignment} \tag{4.9}$$

and the *total column* (TC) value is given by

$$TC = \frac{The\ number\ of\ correclty\ aligned\ columns found\ in\ the\ test\ alignment}{The\ total\ number\ of\ aligned\ columns\ in\ the\ reference\ alignment} \tag{4.10}$$

Table 4.1
Step-by-step calculations of the intermediate values required for the computation of SP and TC

Reference alignment column	**Aligned amino acids pair count in reference alignment**	**Correctly aligned column count in in test alignment**	**Correctly aligned amino acid pair count in test alignment**
1	$\frac{4(4-1)}{2} = 6$	1	$\frac{4(4-1)}{2} = 6$
2	$6 + \frac{4(4-1)}{2} = 12$	2	$6 + \frac{4(4-1)}{2} = 12$
3	$12 + \frac{3(3-1)}{2} = 15$	2	$12 + \frac{2(2-1)}{2} = 13$
4	$15 + \frac{4(4-1)}{2} = 21$	2	$13 + \frac{3(3-1)}{2} = 16$
5	$21 + \frac{3(3-1)}{2} = 24$	2	$16 + \frac{2(2-1)}{2} = 17$
6	$24 + \frac{4(4-1)}{2} = 30$	2	$17 + \frac{3(3-1)}{2} = 20$
7	$30 + \frac{4(4-1)}{2} = 36$	2	$20 + \frac{3(3-1)}{2} = 23$

SP and TC values can be computed by substituting the values given in the last row of Table 4.1 in Equation 4.9 and 4.10

$$SP = \frac{23}{36} = 0.639$$

$$TC = \frac{2}{7} = 0.286$$

■

4.4.1 Evaluation using BAliBASE 3.0

For evaluating multiple sequence alignment algorithms, BAliBASE [32] is the most widely used benchmark. This benchmark contains 3D structure–based alignments that are manually refined. Out of the 386 reference alignments in BAliBASE, there are 186 alignments that have protein sequences which belong to one or the other of the 43 selected protein folds.

The average SP and TC values of MSAIndelFR as well as those of the other six algorithms using this benchmark as reference are shown in Table 4.2. The results show that MSAIndelFR achieves the highest SP and TC values. Specifically, it provides an average SP value of 86.23% representing an improvement of 6.02%, 1.37%, 4.12%, 4.29%, 6.17% and 10.37% over that of MSAProbs, MAFFT, MUSCLE, Clustal Omega, Kalign2 and Clustal W2, respectively. Also, it provides an average TC value of 57.56% representing an improvement of 2.62%, 3.06%, 10.15%, 7.20%, 13.87% and 18.19%, respectively, over that of the six alignment algorithms.

Table 4.2
Average SP and TC values of MSAIndelFR and other multiple alignment algorithms for the benchmarks BAliBASE 3.0, OXBENCH, PREFAB 4.0 and SABmark 1.65

	BAliBASE		OXBENCH		PREFAB		SABmark	
MSA algorithm	SP(%)	TC(%)	SP(%)	TC(%)	SP(%)	TC(%)	SP(%)	TC(%)
MSAIndelFR	**86.23**	**57.56**	**91.88**	**83.83**	**59.35**	**59.35**	**53.59**	**34.38**
MSAProbs	80.21	(54.93)	(89.39)	(79.78)	(57.52)	(57.52)	(51.55)	(25.21)
MAFFT	(84.86)	54.50	88.22	77.98	53.93	53.93	50.14	24.33
MUSCLE	82.11	47.41	88.66	78.93	55.74	55.74	46.33	20.80
Clustal Omega	81.94	50.35	88.05	77.76	55.96	55.96	45.11	19.58
Kalign2	80.06	43.68	87.55	77.30	56.33	56.33	41.64	18.91
Clustal W2	75.86	39.37	87.94	77.00	56.05	56.05	40.38	15.98

Bold faced values indicate the best performance, while the values in parentheses indicate the second best performance.

Boxplots would show more detailed information about the distribution of the SP and TC values than that provided by Table 4.2. They indicate whether a distribution is skewed or whether there are potential unusual observations (outliers) in the data set. In addition, they are very useful when large numbers of test cases are involved and when two or more methods are being compared. Finally, they can be used to determine the first, second (median), and third quartiles as well as interquartile range (IQR) values for various distributions. The width of a box indicates the IQR value, which is the difference between the third and first quartile values.

In view of the above reasons, boxplots resulting from the distributions the SP values of the various algorithms evaluated using BAliBASE 3.0 are shown in Figure 4.2. This figure clearly shows that MSAIndelFR performs better than the other algorithms, since it has the lowest IQR value as well as the highest first quartile value. It is noted that even though MSAIndelFR and MSAprobs have an almost equal median value of 91%, the distribution of the SP values generated by MSAIndelFR is much narrower than that generated by MSAprobs, since the former has an IQR value of 12%, whereas the latter a value of 20%. In addition, it is seen that 75% of the MSAIndelFR alignments have an SP value of more than 84% (first quartile), whereas 25% of the alignments have an SP value of more than 96% (third quartile). Figure 4.3 shows the distributions of the TC values of MSAIndelFR and those of the other six algorithms. It is seen from this figure that, just as the case with respect to the SP values, MSAIndelFR performs better than the other algorithms.

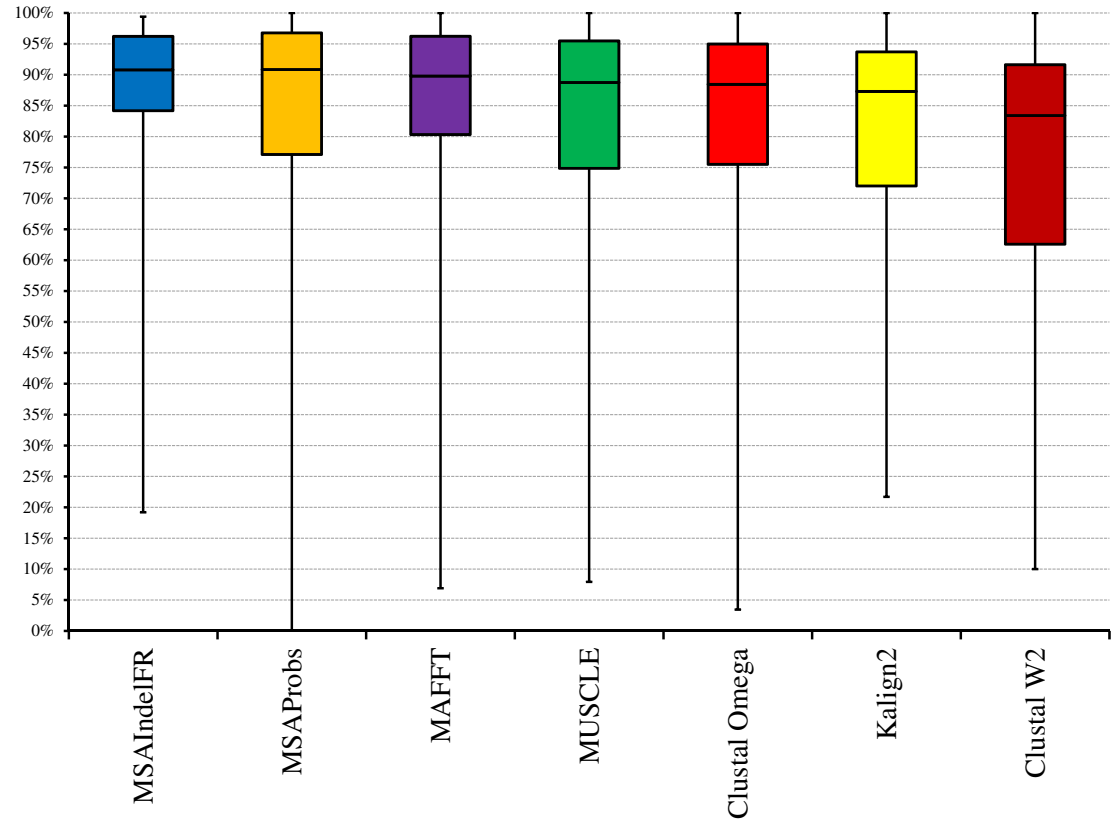


Figure 4.2: Boxplots for the distributions of the SP values of MSAIndelFR and the other MSA multiple alignment algorithms using the BAliBASE 3.0 benchmark, where the top and bottom of a box and the line in between represent the third quartile, first quartile and median, respectively.

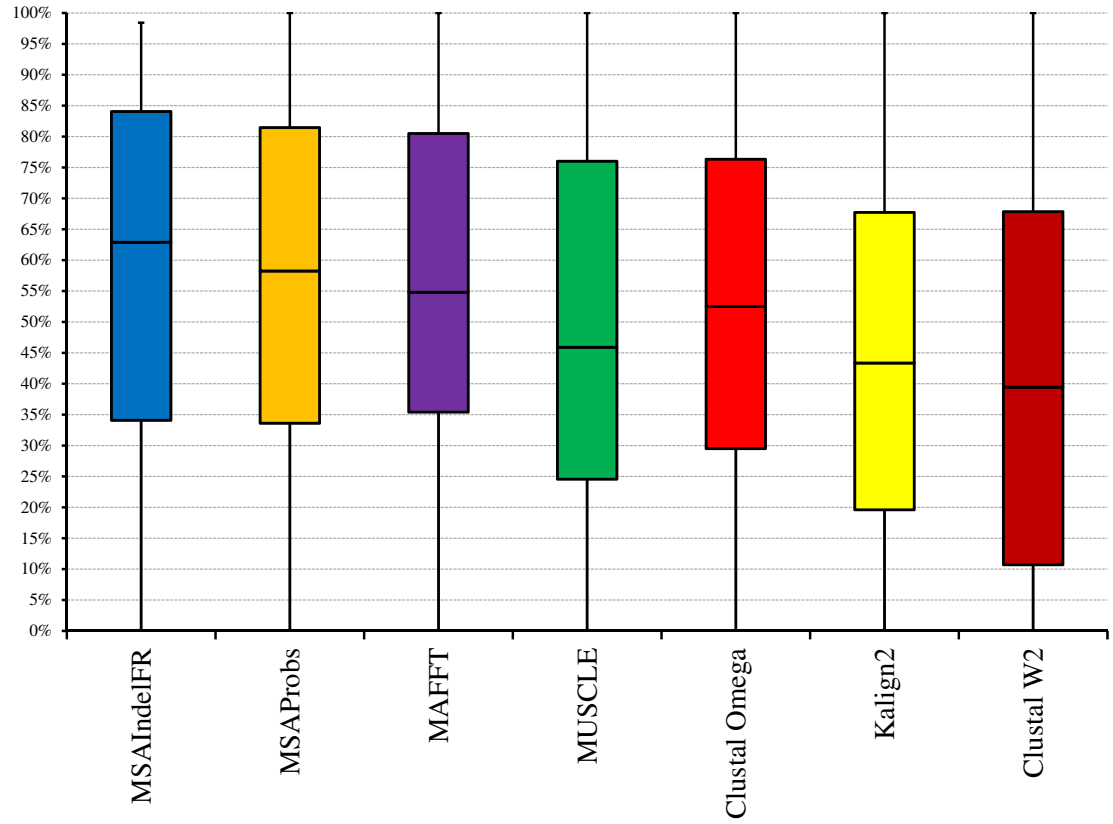


Figure 4.3: Boxplots for the distributions of the TC values of MSAIndelFR and the other MSA multiple alignment algorithms using the BAliBASE 3.0 benchmark, where the top and bottom of a box and the line in between represent the third quartile, first quartile and median, respectively.

4.4.2 Evaluation using OXBENCH

The OXBENCH benchmark [41] is a set of structure-based alignments. Out of the 395 reference alignments in OXBENCH, there are 191 alignments that have protein sequences which belong to one or the other of the 43 selected protein folds.

The average SP and TC values of MSAIndelFR as well as that of the other six algorithms using this benchmark as reference are given in Table 4.2. The results show that MSAIndelFR achieves the highest SP and TC values. Specifically, it provides an average SP value of 91.88% representing an improvement of 2.49%, 3.65%, 3.22%, 3.83%, 4.33% and 3.94% over that of MSAProbs, MAFFT, MUSCLE, Clustal Omega, Kalign2 and Clustal W2, respectively. Also, it provides an average TC value of 83.83% representing an improvement of 4.05%, 5.85%, 4.90%, 6.07%, 6.53% and 6.83%, respectively, over that of the six alignment algorithms.

The boxplots for the distributions of the SP values of the various algorithms evaluated using OXBENCH are shown in Figure 4.4. This figure clearly shows that MSAIndelFR performs better than the other algorithms, since it has the lowest IQR value as well as the highest first quartile value. It is noted that even though MSAIndelFR and MSAprobs have an almost equal median value of 98%, the distribution of the SP values generated by MSAIndelFR is narrower than that generated by MSAprobs, since the former has an IQR value of 9%, whereas the latter an IQR value of 12%. In addition, it is seen that 75% of the MSAIndelFR alignments have an SP value of more than 91% (first quartile), whereas 25% of the alignments have an SP value of 100% (third quartile). The boxplots for the distributions of the TC values of MSAIndelFR and those of the other six algorithms are given in Figure 4.5, which confirm the better performance of MSAIndelFR over the that of other algorithms.

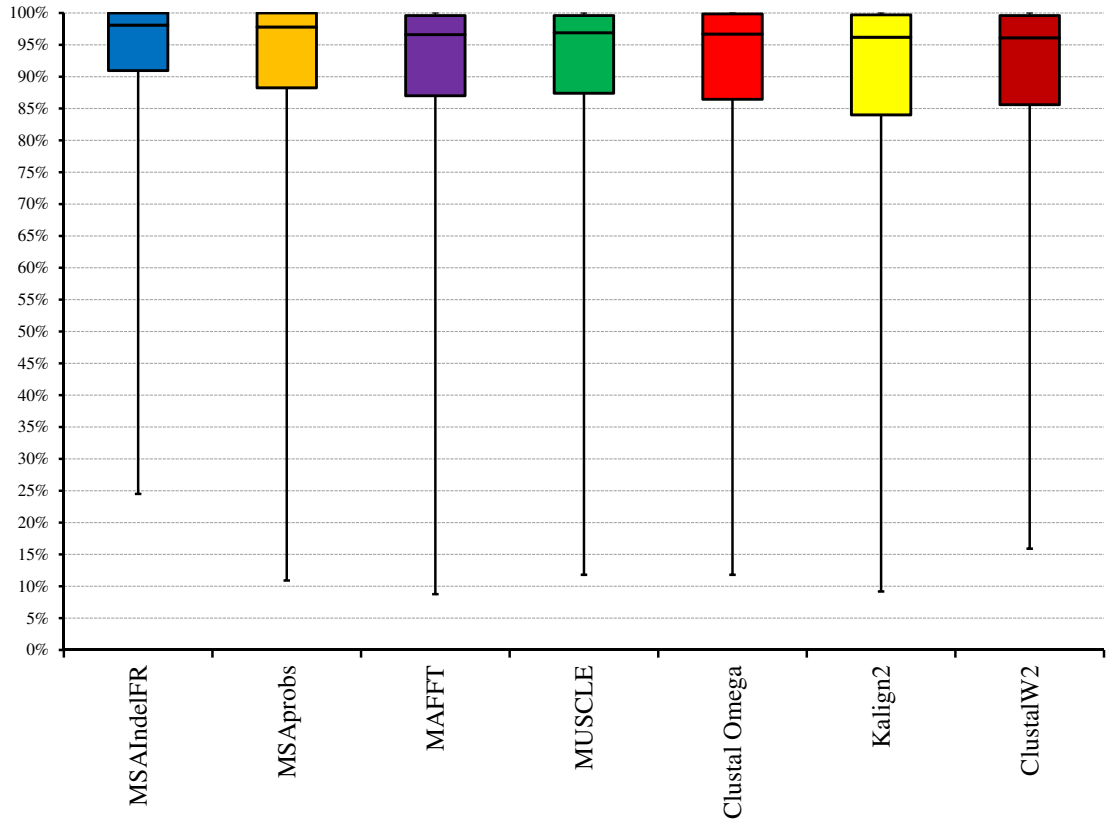


Figure 4.4: Boxplots for the distributions of the SP values of MSAIndelFR and the other MSA algorithms using the OXBENCH benchmark, where the top and bottom of a box and the line in between represent the third quartile, first quartile and median, respectively.

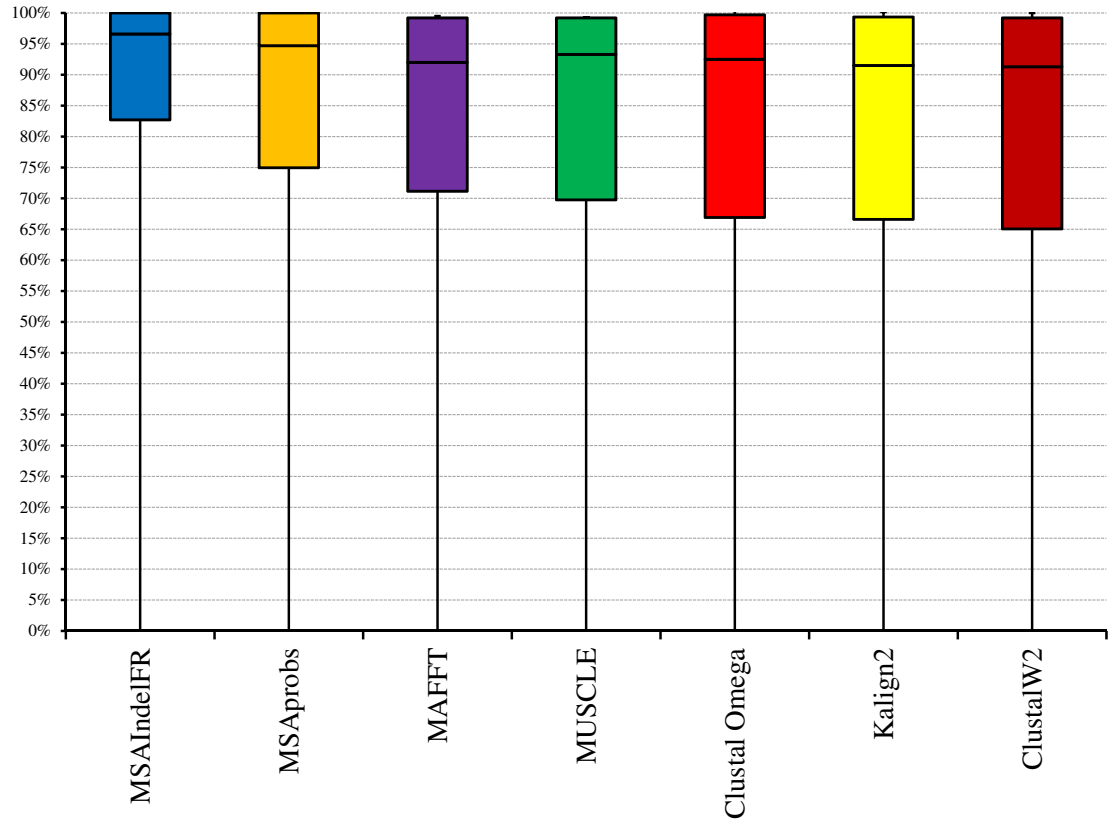


Figure 4.5: Boxplots for the distributions of the TC values of MSAIndelFR and the other MSA algorithms using the OXBENCH benchmark, where the top and bottom of a box and the line in between represent the third quartile, first quartile and median, respectively.

4.4.3 Evaluation using PREFAB 4.0

The PREFAB 4.0 benchmark [15] is a fully automatically–generated benchmark containing 1681 reference alignments. Out of the 1681 reference alignments in PREFAB 4.0, there are 863 alignments that have protein sequences which belong to one or the other of the 43 selected protein folds.

The average SP and TC values of MSAIndelFR as well as those of the other six algorithms using this benchmark as reference are given in Table 4.2. The results show that MSAIndelFR achieves the highest SP and TC values. Specifically, it provides an average SP value of 59.35% representing an improvement of 1.83%, 5.42%, 3.61%, 3.39%, 3.02% and 3.30% over that of MSAProbs, MAFFT, MUSCLE, Clustal Omega, Kalign2 and Clustal W2, respectively. Also, it provides similar improvements in the TC values over that of the other six algorithms.

The boxplots for the distributions of the SP values of the various algorithms are shown in Figure 4.6. This figure clearly shows that MSAIndelFR performs better than the other algorithms, since it has the lowest IQR value as well as the highest first quartile value. It is noted that even though MSAIndelFR, MSAprobs and Clustal W2 have almost the same median value of 67%, the distribution of the SP values generated by MSAIndelFR is narrower than that generated by MSAprobs, since MSAIndelFR has an IQR value of 57%, whereas MSAprobs and Clustal W2 have an IQR value of 61% and 69%, respectively. In addition, it is seen that 75% of the MSAIndelFR alignments have an SP value of more than 31% (first quartile), whereas 25% of the alignments have an SP value of 88% (third quartile). The boxplots for the distributions of the TC values of MSAIndelFR and those of the other six algorithms are shown in Figure 4.7, which confirm the better performance of MSAIndelFR over that of the other algorithms.

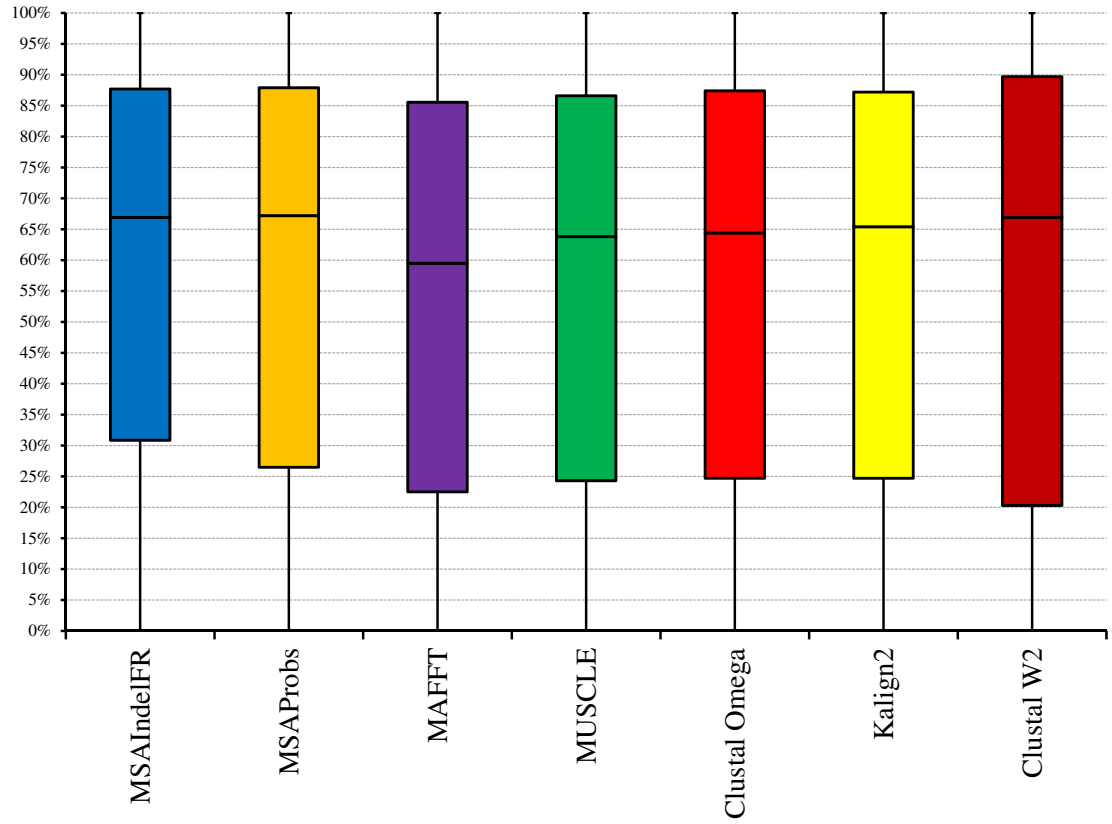


Figure 4.6: Boxplots for the distributions of the SP values of MSAIndelFR and the other MSA algorithms using the PREFAB 4.0 benchmark, where the top and bottom of a box and the line in between represent the third quartile, first quartile and median, respectively.

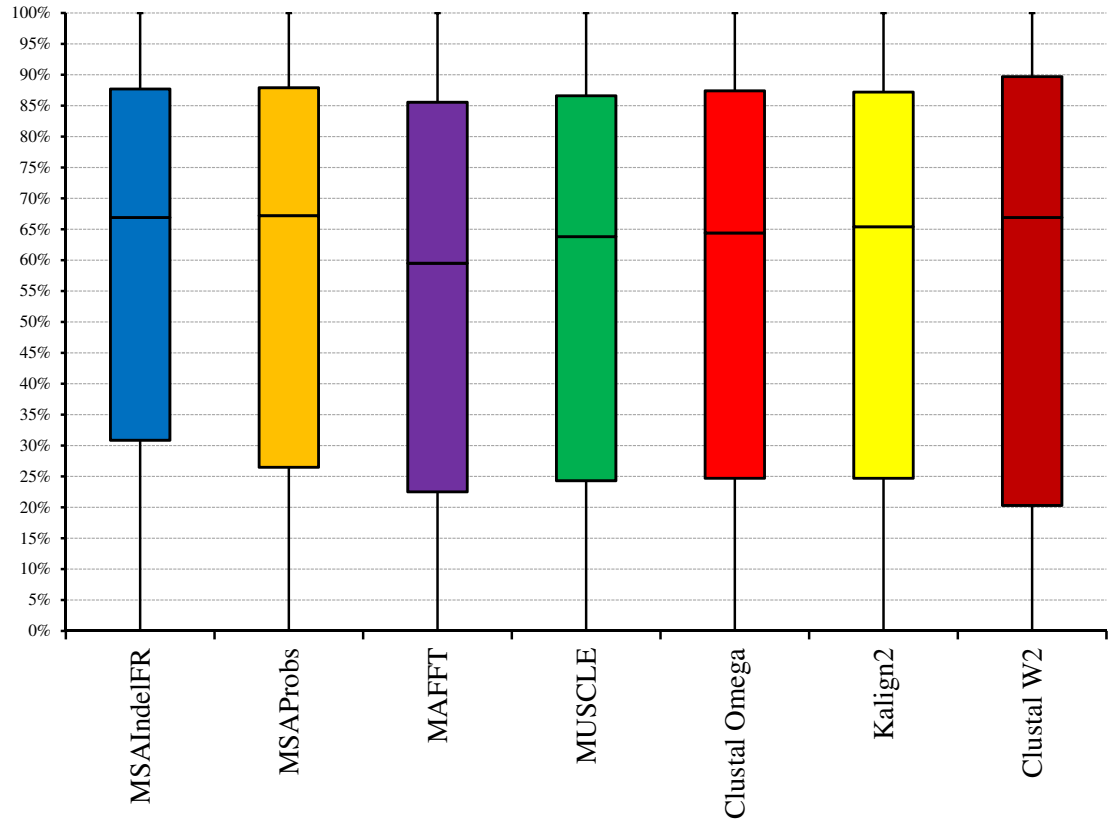


Figure 4.7: Boxplots for the distributions of the TC values of MSAIndelFR and the other MSA algorithms using the PREFAB 4.0 benchmark, where the top and bottom of a box and the line in between represent the third quartile, first quartile and median, respectively.

4.4.4 Evaluation using SABRE (SABmark 1.65)

The SABmark 1.65 [39] is a very challenging benchmark for multiple sequence alignment. This benchmark is divided into two subsets: Twilight zone and Superfamilies. The similarity level between any two protein sequences is less than 50% in the Superfamily set, whereas it is at most 25% in the Twilight set. In [77], the author argued that the pairwise reference alignments in SABmark are not suitable to evaluate MSA algorithms, and hence constructed the SABRE benchmark (*http://www.drive5.com/bench*), containing 423 out of the 634 SABmark groups. In this evaluation, we use SABRE instead of the original SABmark benchmark. Out of the 423 reference alignments in the SABRE benchmark, there are 79 alignments that have protein sequences which belong to one or the other of the 43 selected protein folds.

The average SP and TC values of MSAIndelFR as well as those of the other six algorithms using this benchmark as reference are given in Table 4.2. The results show that MSAIndelFR achieves the highest SP and TC values. Specifically, it provides an average SP value of 53.59% representing an improvement of 2.04%, 3.45%, 7.25%, 8.48%, 11.94% and 13.21% over that of MSAProbs, MAFFT, MUSCLE, Clustal Omega, Kalign2 and Clustal W2, respectively. Also, it provides an average TC value of 34.38% representing an improvement of 9.18%, 10.06%, 13.58%, 14.80%, 15.48% and 18.40%, respectively, over that of the six alignment algorithms.

The boxplots for the distributions of the SP values of the various algorithms are shown in Figure 4.8. This figure clearly shows that even for this challenging benchmark, MSAIndelFR performs better than all the other algorithms in terms of the median value (52%). In addition, it is seen that 75% of MSAIndelFR alignments have an SP value of more than 29% (first quartile), whereas 25% of the alignments have an SP value of more than 77% (third quartile). The boxplots for the distributions of the TC values of MSAIndelFR and those of the other six algorithms are

shown in Figure 4.9, which confirm the better performance of MSAIndelFR over that of the other algorithms.

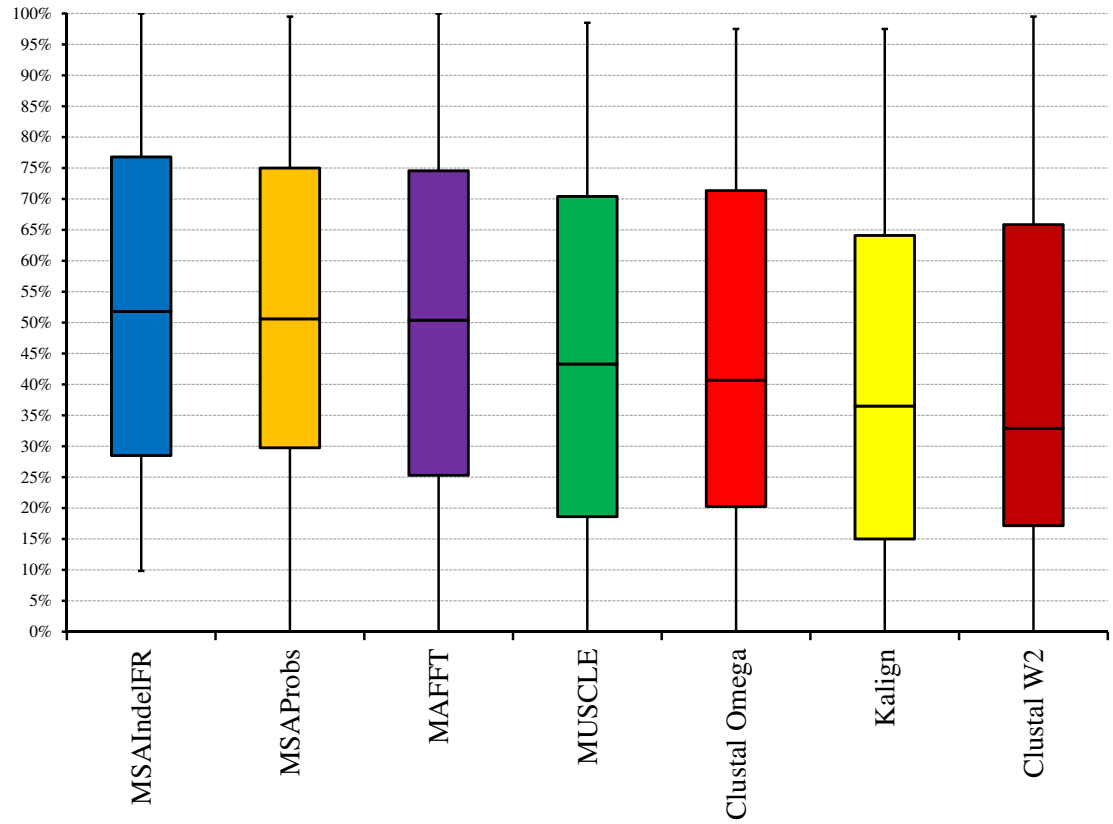


Figure 4.8: Boxplots for the distributions of the SP values of MSAIndelFR and the other algorithms using the SABmark 1.65 benchmark, where the top and bottom of a box and the line in between represent the third quartile, first quartile and median, respectively.

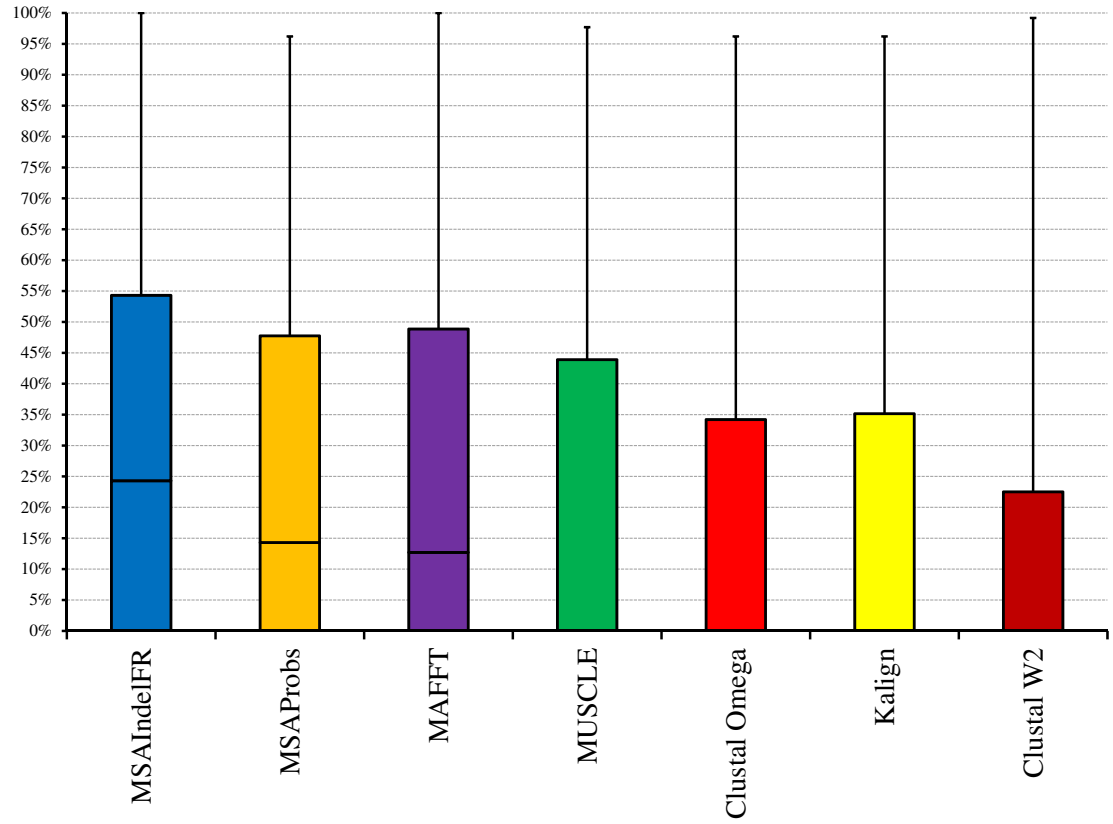


Figure 4.9: Boxplots for the distributions of the TC values of MSAIndelFR and the other algorithms using the SABmark 1.65 benchmark, where the top and bottom of a box and the line in between represent the third quartile, first quartile and median, respectively.

4.5 Summary

In this chapter, a novel and efficient algorithm, called the MSAIndelFR algorithm, has been proposed for multiple protein sequence alignment. In this algorithm, the information on the locations of indel flanking regions and the computed average log–loss values that are obtained from the PPM IndelFR predictor proposed in Chapter 3 have been incorporated. A new variable gap penalty function, wherein the gap opening penalty is position–specific and the gap extension penalty is region–specific, has been proposed. Further, in this algorithm, the so called progressive alignment method for aligning multiple protein sequences has been employed, which in turn has used the dynamic programming approach along with a new variable gap penalty function to align sequences/profiles.

From extensive evaluation results on the four popular benchmarks, namely, BAliBASE 3.0, OXBENCH, PREFAB 4.5, and SABRE (SABmark 1.65), it has been shown that the performance of the proposed MSAIndelFR algorithm is superior to that of the six most–widely used alignment algorithms, Clustal W2, Clustal Omega, MSAProbs, Kalign2, MAFFT and MUSCLE, in terms of both the SP and TC metrics. It is to be pointed out that we have made a study and seen that the sequences chosen from the benchmarks for testing have virtually not been used in the training of the PPM IndelFR predictors [85]. It is also worth pointing out that the improvements in terms of the SP and TC values provided by the proposed MSAIndelFR algorithm have been found [85] to be, in general, statistically significant based the Wilcoxon matched-pair signed-rank test [86]. This superior performance of the proposed algorithm can be attributed to the introduction of the variable gap penalty function, consisting of the position–specific gap opening penalty function and the region-specific gap extension penalty function, in this algorithm.

Finally, we have demonstrated that integrating the information on the computed average log–loss values and the predicted locations of IndelFRs into a multiple se-

quence alignment scheme can substantially improve the accuracy of the alignment.

Chapter 5

Conclusion

5.1 Concluding Remarks

Prediction of indel flanking regions in protein sequences has very important applications in research related to the evolution, structures and functions of proteins, especially with the rapid growth in the number of protein sequences in the protein databases. Generation of a highly accurate multiple protein sequence alignment has important applications in research related to sequence annotation, phylogenetic tree estimation, evolutionary analysis, secondary structure prediction and protein database search. This thesis has been concerned with investigating new techniques for predicting the indel flanking regions (IndelFRs) in protein sequences and with devising a new algorithm for improving the accuracy of multiple protein sequence alignments.

In the first part of this thesis, a novel scheme to predict indel flanking regions in a protein sequence for a given protein fold has been proposed. The proposed *indel flanking region* (IndelFR) predictors, referred to as the PPM IndelFR and PST IndelFR predictors, have been designed based on the *prediction by partial match* (PPM) and *probabilistic suffix tree* (PST), respectively. For the purpose of this

design, the number of protein folds from each of the protein classes *All–α proteins*, *All–β proteins* and *α and β proteins (a/b)* has been chosen for which their indel flanking regions have been specified in the IndelFR database. The proposed IndelFR predictors have been built for various values of the memory length. An algorithm, using the proposed IndelFR predictors, has been developed to identify the locations of IndelFRs in a protein sequence.

The performance of the two proposed IndelFR predictors for the prediction of IndelFRs has been evaluated using the IndelFR database. The evaluation results on this database have shown that the proposed predictors are able to predict IndelFRs in the protein sequences with large values for *accuracy* and F_1-*measure*. The results have also shown that the best choice for the memory length is 4. In order to have a more stringent assessment of the performance of the proposed predictors, they have been tested using a very challenging sequence alignment benchmark, SABmark 1.65. The performance results have shown that the proposed predictors are still able to predict the IndelFRs in the selected protein folds with large values for *accuracy* and F_1-*measure*. The performance of the two proposed predictors have also been compared to that of using the latest version of the alignment software HMMER, HMMER 3.0. The results have shown that from the point of view of predicting IndelFRs in protein sequences, it would be preferable to use the proposed predictors instead of HMMER 3.0 because of the substantially superior performance of the former. It should be noted that the proposed IndelFR predictors are more general than using HMMER 3.0. These predictors for a given protein fold are capable of predicting the indel flanking regions for any protein sequence from any protein family in that fold, whereas HMMER 3.0 has to use different pHMMs depending on the family of the protein fold to which the protein sequence belongs. For instance, for the *Globin-like*, we need to design only one IndelFR predictor fold, while HMMER 3.0 has to use 5 different pHMMs.

In the second part of this thesis, a novel and efficient algorithm, called the MSAIndelFR algorithm, has been proposed for multiple protein sequence alignment. In this algorithm, the information on the indel flanking regions that is obtained from the PPM IndelFR predictor proposed in Chapter 3 has been incorporated. A new *variable gap penalty* function has been introduced, wherein the computed average log–loss values have been used to propose the position–specific gap opening penalty function and the predicted IndelFRs have been used to propose the region–specific gap extension penalty function. In the proposed algorithm, the progressive alignment method for aligning multiple protein sequences has been employed. In the progressive alignment, the new variable gap penalty function has been used in the framework of dynamic programming to conduct a pairwise sequences/profiles alignment.

The performance of the proposed alignment algorithm has been evaluated using four popular benchmarks, BAliBASE 3.0, OXBENCH, PREFAB 4.0, and SABmark 1.65. The results have shown that the performance of MSAIndelFR is superior to that of the six most–widely used alignment algorithms, Clustal W2, Clustal Omega, MSAProbs, Kalign2, MAFFT and MUSCLE, in terms of both the *sum-of-pairs* and *total column* metrics. The results have demonstrated that using the proposed variable gap penalty function based on the computed average log–loss values and the predicted IndelFRs into a multiple sequence alignment algorithm can substantially improve the protein alignment accuracy.

The study undertaken in this thesis has shown that a reliable detection of indels and their flanking regions can be achieved by using the proposed IndelFR predictors, and a substantial improvement in the protein alignment accuracy can be obtained by using the proposed variable gap penalty function that incorporates the information obtained from the use of IndelFR predictors. Thus, we anticipate that our research study will not only enable further studies on the modeling of indel mutations and

on protein sequence alignment, but will also open up new avenues for research concerning evolution, structures, and functions of proteins and their alignments.

5.2 Scope for Future Investigation

The present work can be extended in various ways. Further investigation can be carried out to incorporate other possible characteristic properties of indel flanking regions with a view to enhancing their prediction. For example, the secondary structure information about the indel flanking regions could be incorporated in the proposed IndelFR predictors to enhance its prediction performance. Further investigations can be carried out to incorporate the proposed variable gap penalty function into Hidden Markov Model (HMM) or FFT approach for the alignment of multiple protein sequences.

References

[1] G. Yona, *Introduction to computational proteomics*. Boca Raton: CRC Press, 2011.

[2] N. V. Grishin, "Fold change in evolution of protein structures," *Journal of Structural Biology*, vol. 134, pp. 167–185, May-Jun 2001.

[3] Z. Zhang, J. Huang, Z. Wang, L. Wang, and P. Gao, "Impact of indels on the flanking regions in structural domains," *Molecular Biology and Evolution*, vol. 28, pp. 291–301, Jan 2011.

[4] R. J. Britten, L. Rowen, J. Williams, and R. A. Cameron, "Majority of divergence between closely related dna samples is due to indels," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 100, pp. 4661–4665, Apr 15 2003.

[5] J. Q. Chen, Y. Wu, H. Yang, J. Bergelson, M. Kreitman, and D. Tian, "Variation in the ratio of nucleotide substitution and indel rates across genomes in mammals and bacteria," *Molecular Biology and Evolution*, vol. 26, pp. 1523–1531, Jul 2009.

[6] A. Duval and R. Hamelin, "Mutations at coding repeat sequences in mismatch repair-deficient human cancers: toward a new concept of target genes for instability," *Cancer Research*, vol. 62, pp. 2447–2454, May 1 2002.

[7] M. S. Rosenberg, *Sequence alignment: methods, models, concepts, and strategies*. Univ of California Press, 2009.

[8] A. Lesk, *Introduction to bioinformatics*. Oxford University Press, 2013.

[9] M. A. Larkin, G. Blackshields, N. P. Brown, R. Chenna, P. A. McGettigan, H. McWilliam, F. Valentin, I. M. Wallace, A. Wilm, R. Lopez, J. D. Thompson, T. J. Gibson, and D. G. Higgins, "Clustal W and Clustal X version 2.0," *Bioinformatics*, vol. 23, pp. 2947–2948, Nov 1 2007.

[10] F. Sievers, A. Wilm, D. Dineen, T. J. Gibson, K. Karplus, W. Li, R. Lopez, H. McWilliam, M. Remmert, J. Soding, J. D. Thompson, and D. G. Higgins, "Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega," *Molecular Systems Biology*, vol. 7, p. 539, Oct 11 2011.

[11] T. Lassmann, O. Frings, and E. L. Sonnhammer, "Kalign2: high-performance multiple alignment of protein and nucleotide sequences allowing external features," *Nucleic Acids Research*, vol. 37, pp. 858–865, Feb 2009.

[12] Y. Liu, B. Schmidt, and D. L. Maskell, "MSAProbs: multiple sequence alignment based on pair hidden Markov models and partition function posterior probabilities," *Bioinformatics*, vol. 26, pp. 1958–1964, Aug 15 2010.

[13] K. Katoh, K. ichi Kuma, H. Toh, and T. Miyata, "MAFFT version 5: improvement in accuracy of multiple sequence alignment," *Nucleic Acids Research*, vol. 33, pp. 511–518, January 01 2005.

[14] K. Katoh, K. Misawa, K. Kuma, and T. Miyata, "MAFFT: a novel method for rapid multiple sequence alignment based on fast fourier transform," *Nucleic Acids Research*, vol. 30, pp. 3059–3066, Jul 15 2002.

[15] R. C. Edgar, "MUSCLE: multiple sequence alignment with high accuracy and high throughput," *Nucleic Acids Research*, vol. 32, pp. 1792–1797, March 01 2004.

[16] C. B. Do and K. Katoh, *Protein multiple sequence alignment*, vol. 484 of *Functional Proteomics*, pp. 379–413. Springer, 2008.

[17] Z. Zhang, C. Xing, L. Wang, B. Gong, and H. Liu, "IndelFR: a database of indels in protein structures and their flanking regions," *Nucleic Acids Research*, vol. 40, pp. D512–8, Jan 2012.

[18] Z. Zhang, Y. Wang, L. Wang, and P. Gao, "The combined effects of amino acid substitutions and indels on the evolution of structure within protein families," *PloS one*, vol. 5, no. 12, p. e14316, 2010.

[19] L. Zhu, Q. Wang, P. Tang, H. Araki, and D. Tian, "Genomewide association between insertions/deletions and the nucleotide diversity in bacteria," *Molecular Biology and Evolution*, vol. 26, pp. 2353–2361, Oct 2009.

[20] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison, *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, Jul 1998.

[21] S. Henikoff and J. G. Henikoff, "Amino acid substitution matrices from protein blocks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 89, pp. 10915–10919, Nov 15 1992.

[22] G. H. Gonnet, M. A. Cohen, and S. A. Benner, "Exhaustive matching of the entire protein sequence database," *Science (New York, N.Y.)*, vol. 256, pp. 1443–1445, Jun 5 1992.

[23] M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt, "A model of evolutionary change in proteins," *Atlas of protein sequence and structure*, vol. 5, no. suppl 3, pp. 345–351, 1978.

[24] G. Vogt, T. Etzold, and P. Argos, "An assessment of amino acid exchange matrices in aligning protein sequences: the twilight zone revisited," *Journal of Molecular Biology*, vol. 249, no. 4, pp. 816–831, 1995.

[25] B. Qian and R. A. Goldstein, "Distribution of indel lengths," *Proteins: Structure, Function, and Bioinformatics*, vol. 45, no. 1, pp. 102–104, 2001.

[26] M. S. Chang and S. A. Benner, "Empirical analysis of protein insertions and deletions determining parameters for the correct placement of gaps in protein sequence alignments," *Journal of Molecular Biology*, vol. 341, pp. 617–631, Aug 6 2004.

[27] J. T. Reese and W. R. Pearson, "Empirical determination of effective gap penalties for sequence comparison," *Bioinformatics*, vol. 18, pp. 1500–1507, November 01 2002.

[28] M. A. Zachariah, G. E. Crooks, S. R. Holbrook, and S. E. Brenner, "A generalized affine gap model significantly improves protein sequence alignment accuracy," *Proteins: Structure, Function, and Bioinformatics*, vol. 58, no. 2, pp. 329–338, 2005.

[29] S. F. Altschul, "Generalized affine gap costs for protein sequence alignment," *Proteins: Structure, Function, and Bioinformatics*, vol. 32, no. 1, pp. 88–96, 1998.

[30] I. Miklos, G. A. Lunter, and I. Holmes, "A "long indel" model for evolutionary sequence alignment," *Molecular Biology and Evolution*, vol. 21, pp. 529–540, Mar 2004.

[31] R. A. Cartwright, "Ngila: global pairwise alignments with logarithmic and affine gap costs," *Bioinformatics*, vol. 23, no. 11, pp. 1427–1428, 2007.

[32] J. D. Thompson, P. Koehl, R. Ripp, and O. Poch, "BAliBASE 3.0: latest developments of the multiple sequence alignment benchmark," *Proteins*, vol. 61, pp. 127–136, Oct 1 2005.

[33] J. Pei, "Multiple protein sequence alignment," *Current opinion in structural biology*, vol. 18, no. 3, pp. 382–386, 2008.

[34] J. D. Thompson, F. Plewniak, and O. Poch, "A comprehensive comparison of multiple sequence alignment programs," *Nucleic Acids Research*, vol. 27, pp. 2682–2690, Jul 1 1999.

[35] M. Vingron and M. S. Waterman, "Sequence alignment and penalty choice: Review of concepts, case studies and implications," *Journal of Molecular Biology*, vol. 235, no. 1, pp. 1–12, 1994.

[36] M. Al-Shatnawi, M. O. Ahmad, and M. N. S. Swamy, "Prediction of indel flanking regions in protein sequences using a variable–order Markov model," *Bioinformatics*, vol. 31, pp. 40–47, January 01 2015.

[37] J. Cleary and I. Witten, "Data compression using adaptive coding and partial string matching," *Communications, IEEE Transactions on*, vol. 32, pp. 396–402, 1984.

[38] D. Ron, Y. Singer, and N. Tishby, "The power of amnesia: Learning probabilistic automata with variable memory length," *Machine Learning*, vol. 25, no. 2-3, pp. 117–149, 1996.

[39] I. V. Walle, I. Lasters, and L. Wyns, "SABmark benchmark for sequence alignment that covers the entire known fold space," *Bioinformatics*, vol. 21, pp. 1267–1268, Apr 1 2005.

[40] R. D. Finn, J. Clements, and S. R. Eddy, "HMMER web server: interactive sequence similarity searching," *Nucleic Acids Research*, vol. 39, pp. W29–37, Jul 2011.

[41] G. P. Raghava, S. M. Searle, P. C. Audley, J. D. Barber, and G. J. Barton, "OXBench: a benchmark for evaluation of protein multiple sequence alignment accuracy," *BMC Bioinformatics*, vol. 4, p. 47, Oct 10 2003.

[42] A. Andreeva, D. Howorth, J. M. Chandonia, S. E. Brenner, T. J. Hubbard, C. Chothia, and A. G. Murzin, "Data growth and its impact on the SCOP database: new developments," *Nucleic Acids Research*, vol. 36, pp. D419–25, Jan 2008.

[43] S. A. Benner, M. A. Cohen, and G. H. Gonnet, "Empirical and structural models for insertions and deletions in the divergent evolution of proteins," *Journal of Molecular Biology*, vol. 229, no. 4, pp. 1065–1082, 1993.

[44] M. Hsing and A. Cherkasov, "Indel PDB: a database of structural insertions and deletions derived from sequence alignments of closely related proteins," *BMC Bioinformatics*, vol. 9, p. 293, Jun 25 2008.

[45] S. K. Chan, M. Hsing, F. Hormozdiari, and A. Cherkasov, "Relationship between insertion/deletion (indel) frequency of proteins and essentiality," *BMC Bioinformatics*, vol. 8, p. 227, Jun 28 2007.

[46] H. Jiang and C. Blouin, "Insertions and the emergence of novel protein structure: a structure-based phylogenetic study of insertions," *BMC Bioinformatics*, vol. 8, p. 444, Nov 15 2007.

[47] M. Nisenson, I. Yariv, R. El-Yaniv, and R. Meir, *Towards behaviometric security systems: Learning to identify a typist*, pp. 363–374. Knowledge Discovery in Databases: PKDD 2003, Springer, 2003.

[48] F. M. J. Willems, Y. M. Shtarkov, and T. J. Tjalkens, "The context-tree weighting method: basic properties," *Information Theory, IEEE Transactions on*, vol. 41, no. 3, pp. 653–664, 1995.

[49] J. Rissanen, "A universal data compression system," *Information Theory, IEEE Transactions on*, vol. 29, no. 5, pp. 656–664, 1983.

[50] R. Begleiter, R. El-Yaniv, and G. Yona, "On prediction using variable order Markov models," *Journal of Artificial Intelligence Research*, pp. 385–421, 2004.

[51] G. Bejerano and G. Yona, "Variations on probabilistic suffix trees: statistical modeling and prediction of protein families," *Bioinformatics*, vol. 17, pp. 23–43, 2001.

[52] A. Krogh, M. Brown, I. S. Mian, K. Sjlander, and D. Haussler, "Hidden Markov models in computational biology. applications to protein modeling," *Journal of Molecular Biology*, vol. 235, no. 5, pp. 1501–1531, 1994.

[53] S. R. Eddy, "Profile hidden Markov models," *Bioinformatics*, vol. 14, no. 9, pp. 755–763, 1998.

[54] R. Hughey and A. Krogh, "SAM: Sequence alignment and modeling software system," *Technical Report, UCSC-CRL-95-7, University of California, Santa Cruz, Computer Engineering, UC Santa Cruz, CA.*, 1995.

[55] K. Karplus, C. Barrett, and R. Hughey, "Hidden Markov models for detecting remote protein homologies," *Bioinformatics*, vol. 14, no. 10, pp. 846–856, 1998.

[56] M. Punta, P. C. Coggill, R. Y. Eberhardt, J. Mistry, J. Tate, C. Boursnell, N. Pang, K. Forslund, G. Ceric, J. Clements, A. Heger, L. Holm, E. L. Sonnhammer, S. R. Eddy, A. Bateman, and R. D. Finn, "The Pfam protein families database," *Nucleic Acids Research*, vol. 40, pp. D290–301, Jan 2012.

[57] C. Notredame, "Recent progress in multiple sequence alignment: a survey," *Pharmacogenomics*, vol. 3, no. 1, pp. 131–144, 2002.

[58] K. Katoh and D. M. Standley, "MAFFT multiple sequence alignment software version 7: improvements in performance and usability," *Molecular Biology and Evolution*, vol. 30, pp. 772–780, Apr 2013.

[59] R. C. Edgar, "MUSCLE: multiple sequence alignment with high accuracy and high throughput," *Nucleic Acids Research*, vol. 32, pp. 1792–1797, Mar 19 2004.

[60] D. F. Feng and R. F. Doolittle, "Progressive sequence alignment as a prerequisite to correct phylogenetic trees," *Journal of Molecular Evolution*, vol. 25, no. 4, pp. 351–360, 1987.

[61] J. D. Thompson, D. G. Higgins, and T. J. Gibson, "Clustal W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice," *Nucleic Acids Research*, vol. 22, pp. 4673–4680, Nov 11 1994.

[62] W. J. Wilbur and D. J. Lipman, "Rapid similarity searches of nucleic acid and protein data banks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 80, pp. 726–730, Feb 1983.

[63] N. Saitou and M. Nei, "The neighbor-joining method: a new method for reconstructing phylogenetic trees," *Molecular Biology and Evolution*, vol. 4, pp. 406–425, Jul 1987.

[64] E. W. Myers and W. Miller, "Optimal alignments in linear space," *Computer applications in the biosciences : CABIOS*, vol. 4, pp. 11–17, March 01 1988.

[65] R. R. Sokal, "A statistical method for evaluating systematic relationships," *Univ Kans Sci Bull*, vol. 38, pp. 1409–1438, 1958.

[66] T. Lassmann and E. L. Sonnhammer, "Kalign–an accurate and fast multiple sequence alignment algorithm," *BMC Bioinformatics*, vol. 6, p. 298, Dec 12 2005.

[67] S. Wu and U. Manber, "Fast text searching: allowing errors," *Communications of the ACM*, vol. 35, no. 10, pp. 83–91, 1992.

[68] R. Muth and U. Manber, "Approximate multiple string search," in *Proceedings of the7th Annual Symposium on Combinatorial Pattern*, vol. 1075, pp. 75–86, Springer, 1996.

[69] W. R. Taylor, *Protein structure comparison using SAP*, pp. 19–32. Protein structure prediction, Springer, 2000.

[70] R. B. Russell and G. J. Barton, "Multiple protein sequence alignment from tertiary structure comparison: assignment of global and residue confidence levels," *Proteins: Structure, Function, and Bioinformatics*, vol. 14, no. 2, pp. 309–323, 1992.

[71] A. S. Siddiqui, U. Dengler, and G. J. Barton, "3Dee: a database of protein structural domains," *Bioinformatics*, vol. 17, pp. 200–201, Feb 2001.

[72] L. Holm and C. Sander, "Touring protein fold space with Dali/FSSP," *Nucleic Acids Research*, vol. 26, pp. 316–319, January 01 1998.

[73] I. N. Shindyalov and P. E. Bourne, "Protein structure alignment by incremental combinatorial extension (CE) of the optimal path.," *Protein engineering*, vol. 11, pp. 739–747, September 01 1998.

[74] S. F. Altschul, T. L. Madden, A. A. Schffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, "Gapped BLAST and PSI–BLAST: a new generation of protein database search programs," *Nucleic Acids Research*, vol. 25, pp. 3389–3402, September 01 1997.

[75] I. V. Walle, I. Lasters, and L. Wyns, "Align-m–a new algorithm for multiple alignment of highly divergent sequences," *Bioinformatics*, vol. 20, pp. 1428–1435, Jun 12 2004.

[76] N. S. Boutonnet, M. J. Rooman, M. E. Ochagavia, J. Richelle, and S. J. Wodak, "Optimal protein structure alignments by multiple linkage clustering: application to distantly related proteins," *Protein engineering*, vol. 8, pp. 647–662, Jul 1995.

[77] R. C. Edgar, "Quality measures for protein alignment benchmarks," *Nucleic Acids Research*, vol. 38, pp. 2145–2153, April 01 2010.

[78] P. Buhlmann and A. J. Wyner, "Variable length Markov chains," *The Annals of Statistics*, vol. 27, no. 2, pp. 480–513, 1999.

[79] A. Moffat, "Implementing the PPM data compression scheme," *Communications, IEEE Transactions on*, vol. 38, no. 11, pp. 1917–1921, 1990.

[80] J. G. Henikoff and S. Henikoff, "Using substitution probabilities to improve position-specific scoring matrices," *Computer applications in the biosciences : CABIOS*, vol. 12, pp. 135–143, Apr 1996.

[81] T. Fawcett, "An introduction to ROC analysis," *Pattern Recogn.Lett.*, vol. 27, pp. 861–874, jun 2006.

[82] P. Sonego, A. Kocsor, and S. Pongor, "ROC analysis: applications to the classification of biological sequences and 3D structures," *Briefings in Bioinformatics*, vol. 9, pp. 198–209, May 2008.

[83] N. Chinchor, "MUC-4 evaluation metrics," in *Proceedings of the 4th conference on Message understanding*, MUC4 '92, (Stroudsburg, PA, USA), pp. 22–29, Association for Computational Linguistics, 1992.

[84] M. Punta *et al.*, "The Pfam protein families database," *Nucleic Acids Research*, vol. 40, pp. D290–D301, 2012.

[85] M. Al-Shatnawi, M. O. Ahmad, and M. N. S. Swamy, "MSAIndelFR: A scheme for multiple protein sequence alignment using information on Indel flanking regions," under second round of reviews by *BMC Bioinformatics*.

[86] F. Wilcoxon, "Probability tables for individual comparisons by ranking methods," *Biometrics*, vol. 3, pp. 119–122, 1947.

Appendix

Table A.1
Protein folds from the *All-α protein* class that are listed in the IndelFR database, and their Pfam families

label	**Protein folds**	**Protein superfamilies**	**Pfam families**
A1	a.1: Globin–like	a.1.1: Globin–like	Globin (PF00042)
			Phycobilisome (PF00502)
			FAD_binding_2 (PF00890)
			Bac_globin (PF01152)
			Dus (PF01207)
A3	a.3: Cytochrome c	a.3.1: Cytochrome c	Cytochrom_B_C (PF00032)
			Cytochrom_C (PF00034)
A4	a.4: DNA/RNAbinding 3–helical bundle	a.4.1: Homeodomain–like a.4.5: "Winged helix" DNA-binding domain	Homeobox (PF00046)
			TetR_N (PF00440)
			MarR (PF01047)
			HxlR (PF01638)
			TrmB (PF01978)
			HTH_5 (PF01022)
			HTH_8 (PF02954)
			PadR (PF03551)

(Continued on next page)

Table A.1 – (Continued from previous page)

label	Protein folds	Protein superfamilies	Pfam families
			Myb_DNA–bind_6 (PF13921)
A22	a.22: Histone–fold	a.22.1: Histone–fold	Histone (PF00125)
			TBP (PF00352)
A25	a.25: Ferritin–like	a.25.1: Ferritin–like	Ferritin (PF00210)
			Ribonuc_red_sm (PF00268)
			Phenol_Hydrox (PF02332)
A26	a.26: 4–helical cytokines	a.26.1: 4–helical cytokines	Hormone_1 (PF00103)
			IL10 (PF00726)
			EPO_TPO (PF00758)
			LIF_OSM (PF01291)
A35	a.35: lambda repressor–like DNA–binding domains	a.35.1: lambda repressor–like DNA-binding domains	HTH_3 (PF01381)
A39	a.39: EF Hand–like	a.39.1: EF–hand	Efhand (PF00036)
A45	a.45: Glutathione S–transferase (GST), C–terminal domain	a.45.1: Glutathione S–transferase (GST), C–terminal domain	GST_C (PF00043)
			GST_N (PF02798)
A118	a.118: alpha–alpha superhelix	a.118.1: ARM repeat	14–3–3 (PF00244)
		a.118.3: Sec7 domain	Arm (PF00514)

(Continued on next page)

Table A.1 – (Continued from previous page)

label	Protein folds	Protein superfamilies	Pfam families
		a.118.7: 14-3-3 protein	VHS (PF00790)
		a.118.9: ENTH/VHS domain	Sec7 (PF01369)
		a.118.8: TPR–like	ENTH (PF01417)
			TPR_2 (PF07719)
A133	a.133: Phospholipase A2, PLA2	a.133.1: Phospholipase A2, PLA2	Phospholip_A2_1(PF00068)

Table A.2
Protein folds from the *All-β protein* class that are listed in the IndelFR database, and their Pfam families

label	**Protein folds**	**Protein superfamilies**	**Pfam families**
B6	b.6: Cupredoxin–like	b.6.1: Cupredoxins	COX2 (PF00116)
			Copper–bind (PF00127)
			Cu–oxidase (PF00394)
B18	b.18: Galactose–binding domain-like	b.18.1: Galactose–binding domain-like	F5_F8_type_C (PF00754)
			CBM_6 (PF03422)
B29	b.29: Concanavalin A–like lectins/glucanases	b.29.1: Concanavalin A–like lectins/glucanases	Laminin_G_1 (PF00054)
			Lectin_legB (PF00139)
			Gal–bind_lectin (PF00337)
			Glyco_hydro_11 (PF00457)
			Glyco_hydro_7 (PF00840)
			Glyco_hydro_12 (PF01670)
B34	b.34: SH3–like barrel	b.34.2: SH3–domain	SH3_1 (PF00018)
			MBT (PF02820)
B35	b.35: GroES–like	b.35.1: GroES–like	Cpn10 (PF00166)

(Continued on next page)

Table A.2 – (Continued from previous page)

label	Protein folds	Protein superfamilies	Pfam families
			ADH_zinc_N (PF00107)
B36	b.36: PDZ domain–like	b.36.1: PDZ domain–like	PDZ (PF00595)
B40	b.40: OB–fold	b.40.5: Inorganic pyrophosphatase	tRNA–synt_2 (PF00152)
		b.40.1: Staphylococcal nuclease	CSD (PF00313)
		b.40.6: MOP–like	SSB (PF00436)
			SNase (PF00565)
			Pyrophosphatase (PF00719)
			tRNA_bind (PF01588)
			Stap_Strp_tox_C (PF02876)
B42	b.42: beta–Trefoil	b.42.1: Cytokine	FGF (PF00167)
		b.42.2: Ricin B–like lectins	Ricin_B_lectin (PF00652)
B47	b.47: Trypsin–like serine proteases	b.47.1: Trypsin–like serine proteases	Trypsin (PF00089)
			Peptidase_C3 (PF00548)
			Trypsin_2 (PF13365)
B50	b.50: Acid proteases	b.50.1: Acid proteases	Asp (PF00026)
			RVP (PF00077)
B55	b.55: PH domain–like	b.55.1: PH domain–like	PH (PF00169)

(Continued on next page)

Table A.2 – (Continued from previous page)

label	Protein folds	Protein superfamilies	Pfam families
			WH1 (PF00568)
			PID (PF00640)
B60	b.60: Lipocalins	b.60.1: Lipocalins	Lipocalin (PF00061)
			DUF1794 (PF08768)
B82	b.82: Double–stranded beta–helix	b.82.1: RmlC–like cupins b.82.3: cAMP–binding domain–like	cNMP_binding (PF00027)
			Cupin_1 (PF00190)
			dTDP_sugar_isom (PF00908)
			Cupin_2 (PF07883)
B121	b.121: Nucleoplasmin–like/VP (viral coat and capsid proteins)	b.121.2: Group II dsDNA viruses VP b.121.4: Positive stranded ssRNA viruses	Rhv (PF00073)
			Viral_coat (PF00729)
			Parvo_coat (PF00740)
			Tymo_coat (PF00983)
			Peptidase_A6 (PF01829)
			Phage_F (PF02305)

Table A.3
Protein folds from the *α and β proteins (a/b)* class that are listed in the IndelFR database, and their Pfam families

label	**Protein folds**	**Protein superfamilies**	**Pfam families**
C1	c.1: TIM beta/alpha-barrel	c.1.1: Triosephosphate isomerase (TIM)	TIM (PF00121)
		c.1.2: Ribulose–phoshate binding barrel	Alpha–amylase (PF00128)
		c.1.3: Thiamin phosphate synthase	Cellulase (PF00150)
		c.1.4: FMN–linked oxidoreductases	Glyco_hydro_1 (PF00232)
		c.1.5: Inosine monophosphate dehydrogenase (IMPDH)	Aldo_ket_red (PF00248)
		c.1.6: PLP-binding barrel	Glycolytic (PF00274)
		c.1.7: NAD(P)-linked oxidoreductase	Glyco_hydro_18 (PF00704)
		c.1.8: (Trans)glycosidases	
		c.1.9: Metallo-dependent hydrolases	
		c.1.10: Aldolase	
		c.1.11: Enolase C-terminal domain-like	
		c.1.12: Phosphoenolpyruvate	

(Continued on next page)

Table A.3 – (Continued from previous page)

label	Protein folds	Protein superfamilies	Pfam families
		/pyruvate domain	
		c.1.13: Malate synthase G	
		c.1.14: RuBisCo, C-terminal domain	
		c.1.15: Xylose isomerase-like	
		c.1.16: Bacterial luciferase-like	Oxidored_FMN (PF00724)
		c.1.17: Nicotinate	
		/Quinolinate PRTase	
		C-terminal domain-like	MR_MLE (PF01188)
		c.1.18: PLC-like phosphodiesterases	AP_endonuc_2 (PF01261)
		c.1.19: Cobalamin (vitamin B12)	
		-dependent enzymes	RuBisCO_large_N (PF02788)
		c.1.20: tRNA-guanine transglycosylase	Enolase_N (PF03952)
		c.1.21: Dihydropteroate synthetase	
		-like	Amidohydro_3 (PF07969)
		c.1.26: Homocysteine	
		S-methyltransferase	
		c.1.27: (2r)–phospho-3	

(Continued on next page)

Table A.3 – (Continued from previous page)

label	Protein folds	Protein superfamilies	Pfam families
		-sulfolactate synthase ComA	FMN_dh (PF01070)
C2	c.2: NAD(P)–binding Rossmann-fold domains	c.2.1: NAD(P)–binding Rossmann–fold domains	Gp_dh_N (PF00044)
			Ldh_1_N (PF00056)
			adh_short (PF00106)
			ADH_zinc_N (PF00107)
			THF_DHG_CYH (PF00763)
			NAD_Gly3P_dh_N (PF01210)
			NAD_binding_2 (PF03446)
C3	c.3: FAD/NAD(P) -binding domain	c.3.1: FAD/NAD(P) -binding domain	Pyr_redox (PF00070)
			FAD_binding_2 (PF00890)
			DAO (PF01266)
C14	c.14: ClpP/crotonase	c.14.1: ClpP/crotonase	ECH (PF00378)
			CLP_protease (PF00574)
			Carboxyl_trans (PF01039)
		c.23.1: CheY-like	Response_reg (PF00072)
		c.23.5: Flavoproteins	GATase (PF00117)
		c.23.6: Cobalamin (vitamin B12)	

(Continued on next page)

Table A.3 – (Continued from previous page)

label	Protein folds	Protein superfamilies	Pfam families
C23	c.23: Flavodoxin–like	–binding domain	Flavodoxin_1 (PF00258)
		c.23.8: N5–CAIR mutase (phosphoribosy–laminoimidazole carboxylase, PurE)	DJ-1_PfpI (PF01965)
		c.23.12: Formate/glycerate dehydrogenase catalytic domain-like	FMN_red (PF03358)
		c.23.13: Type II 3–dehydroquinate dehydratase	Flavodoxin_2 (PF02525)
		c.23.14: N–(deoxy) ribosyltransferase–like	
		c.23.16: Class I glutamine amidotransferase-like	
C26	c.26: Adenine nucleotide alpha hydrolase–like	c.26.1: Nucleotidylyl transferase	tRNA–synt_1 (PF00133)
			ETF (PF01012)
		c.26.2: Adenine nucleotide alpha hydrolases–like	CTP_transf_2 (PF01467)
			NAD_synthase (PF02540)
			TPP_enzyme_M (PF00205)

(Continued on next page)

Table A.3 – (Continued from previous page)

label	Protein folds	Protein superfamilies	Pfam families
C36	c.36: Thiamin diphosphate -binding fold (THDP-binding)	c.36.1: Thiamin diphosphate -binding fold (THDP-binding)	Transketolase_N (PF00456)
			E1_dh (PF00676)
			TPP_enzyme_N (PF02776)
			Transket_pyr (PF02779)
C37	c.37: P-loop containing nucleoside triphosphate hydrolases	c.37.1: P-loop containing nucleoside triphosphate hydrolases	AAA (PF00004)
			ABC_tran (PF00005)
			GTP_EFTU (PF00009)
			Arf (PF00025)
			RecA (PF00154)
			Kinesin (PF00225)
			DEAD (PF00270)
			Helicase_C (PF00271)
			ADK (PF00406)
			Sulfotransfer_1 (PF00685)
			MMR_HSR1 (PF01926)
			AAA_2 (PF07724)

(Continued on next page)

Table A.3 – (Continued from previous page)

label	Protein folds	Protein superfamilies	Pfam families
			AAA_5 (PF07728)
C47	c.47: Thioredoxin fold	c.47.1: Thioredoxin–like	GST_C (PF00043)
			Thioredoxin (PF00085)
			AhpC–TSA (PF00578)
			GST_N (PF02798)
			Redoxin (PF08534)
			Thioredoxin_2 (PF13098)
			GST_N_3 (PF13417)
			Thioredoxin_8 (PF13905)
C55	c.55: Ribonuclease H-like motif	c.55.1: Actin-like ATPase domain	HSP70 (PF00012)
		c.55.2: Creatinase/prolidase N-terminal domain	ROK (PF00480)
			Peptidase_M24 (PF00557)
		c.55.3: Ribonuclease H-like	rve (PF00665)
			DNA_pol_B_exo2 (PF10108)
C56	c.56: Phosphorylase/hydrolase-like	c.56.2: Purine and uridine phosphorylases	PNP_UDP_1 (PF01048)
			Peptidase_M20 (PF01546)
		c.56.5: Zn-dependent	Propep_M14 (PF02244)

(Continued on next page)

Table A.3 – (Continued from previous page)

label	Protein folds	Protein superfamilies	Pfam families
		exopeptidases	M20_dimer (PF07687)
C61	c.61: PRTase–like	c.61.1: PRTase–like	Pribosyltran (PF00156)
C67	c.67: PLP–dependent transferase–like	c.67.1: PLP-dependent transferases	Aminotran_1_2 (PF00155)
			Cys_Met_Meta_PP (PF01053)
			Beta_elim_lyase (PF01212)
C68	c.68: Nucleotide–diphospho –sugar transferases	c.68.1: Nucleotide–diphospho –sugar transferases	Hexapep (PF00132)
			NTP_transferase (PF00483)
			CTP_transf_3 (PF02348)
C69	c.69: alpha/beta–Hydrolases	c.69.1: alpha/beta–Hydrolases	Abhydrolase_1 (PF00561)
			Esterase (PF00756)
			Abhydrolase_3 (PF07859)
			Abhydrolase_5 (PF12695)
			Abhydrolase_6 (PF12697)
C94	c.94: Periplasmic binding protein-like II	c.94.1: Periplasmic binding protein–like II	Transferrin (PF00405)
			LysR_substrate (PF03466)
	c.95: Thiolase–like	c.95.1: Thiolase–like	Thiolase_N (PF00108)
			Chal_sti_synt_N (PF00195)

(Continued on next page)

Table A.3 – (Continued from previous page)

label	**Protein folds**	**Protein superfamilies**	**Pfam families**
C95			Chal_sti_synt_C (PF02797)
			Thiolase_C (PF02803)
C108	c.108: HAD-like	c.108.1: HAD-like	Hydrolase (PF00702)
			Hydrolase_3 (PF08282)
			HAD (PF12710)
			HAD_2 (PF13419)