ZURICH UNIVERSITY OF APPLIED SCIENCES

Master of Science (MSc) ZFH in Life Sciences with specialisation in Applied Computational Life Sciences

Zürcher Hochschule
für Angewandte Wissenschaften

Master's Degree Thesis

A study of dynamics of Indels using ProPIP, PRANK and MAFFT

Supervisors

Dr.MANUEL GIL

Dr.MARIA ANISIMOVA

Dr.MASSIMO MAIOLO

Candidate

ELDHOSE POULOSE

JULY 2020

Acknowledgements

This thesis represents not only my work at ZHAW, it is a milestone of my entire life. My life at ZHAW has been an amazing journey with many ups and downs. This thesis presents the lessons I learned from one of the tools created by the Applied Computational Genomics Team, ZHAW: ProPIP. ProPIP is an end result of the work by a group of people, who I wish to acknowledge.

First and foremost, I would like to express my deepest appreciation to my supervisor, professor **Dr. Manuel Gil**. Manuel helped me throughout the Master's programme to design my entire course and he has supported me not only by providing guidance for the thesis topic, but also mentally and emotionally through the rough road to finish the thesis. I would also like to extend my deepest gratitude to **Dr. Maria Anisimova**, who is the head of our team. Maria was the first person who welcomed me on my first day at ZHAW. Even though I was mostly in contact with Manuel, Maria generously helped me in solving my tasks and corrected me when I misinterpret different scientific terms. She has been a wonderful energetic supervisor during my thesis work. Furthermore, I'm deeply indebted to **Dr. Massimo Maiolo** (Max). The discussion days with Max has been full of knowledge and experience. Max provided many important inputs and suggestions to this work. His extensive support and help throughout this short period of thesis workdays actually eased me to get some interesting results at the end.

Moreover, I am grateful to my colleagues **Hossein and Matteo** for helping me to solve some problem during the initial days of my thesis work at Schloss, Wädenswil. I also had great pleasure of working with **Jithin and Phuong**, together with whom I did the preliminary studies related to ProPIP. Last but not least, I would like to extend my sincere thanks to the HPC cluster team at ZHAW. Especially **Geert**, who figured out the problem with my Cluster account during the very first week of my thesis.

The people from Catholic church Wädenswil is like my family in Switzerland, which is where I have my living space. I praise the enormous amount of help and support by the priest, **Markus Dettling** throughout these years, who has been an inspiration and a positive energy for me to accept the different situations of life with a positive mindset. I finish with Kerala, a southern state in India, where the most basic source of my life energy resides: my parents. I have an amazing family, unique in many ways. Their support has been unconditional these years; they have given up many things for me to be here at ZHAW. Finally, I thank God for providing me the courage and, being my strength throughout my entire life in Switzerland. Amen!

Abstract

Evolutionary changes happens over time in the genomes of species. Two of the most important mechanisms of evolutionary changes are substitutions and indels (insertions and deletions). Analyses of genomic sequences typically rely on multiple sequence alignments (MSA), which infer indels. The information regarding the ancestral sequences are limited and contain more events such as insertion, deletion and substitution. To mimic the natural selection of nature where short extinctions (short events) and mass extinctions (long events) exist the simulation based methods are commonly used and these natural events represents, for instance, the insertions, deletions or substitutions of genomic residues. Multitude of methods and estimators exist to do the MSA inference problem. ProPIP is an MSA estimator developed by the Applied Computational Genomics Team (ACGT) at Institute of Applied Simulations (IAS), ZHAW. The ProPIP relies on an explicit evolutionary model of indel (as opposed to more traditional aligners) termed Poisson Indel Process (PIP) [2]. This thesis focus on the suitability of PIP (which is a single character indel model) to infer long indels and the performance our aligner ProPIP to other traditional aligners.

To address the objectives of this thesis, both simulated as well as real data were used. The simulated data consist of single indel and long indel data. The single indel data was simulated according to PIP model using the simulator named PIPJava provided by the authors of PIP, Alexandre Bouchard and Michael I. Jordan [2]. And, the long indel data was simulated using the widely used simulator called INDELible v1.03 created by William Fletcher and Ziheng Yang [6]. To include real data into this study, the most frequently studied cluster of gene family named γ-Proteobacteria is considered. For this, the protein alignments used in the phylogeny reconstruction experiment conducted by Nancy A. Moran [13] is utilized.

Multiple sequences from simulated as well as real data were reconstructed using our aligner ProPIP and two state of the art aligners PRANK v.170427 (a phylogeny-aware alignment algorithm based aligner) and MAFFT v7.453 (a Fast Fourier Transform (FFT) based aligner). Additionally, to test the accuracy of our aligner ProPIP, we used the supplementary features from our tool. Also, to tune the indel placements, a new method (k-Factor) is introduced and tested in ProPIP. After the MSA reconstruction, from each sequence on each MSA's, an indel length (ref: Section 5.2) and an indel block (ref: Section 5.3) method is applied to retrieve the information about the gaps. Later, the difference in the number of gaps between the "true" MSA and the inferred MSA is assessed using indel length distribution method and indel block distribution method (explained in Chapter 5). Furthermore, the distance to "true" MSA is calculated using classical quality scoring mechanism qscore [19], which calculates four different quality scores from the "true" MSAs (simulated and real) and inferred MSAs (by ProPIP, PRANK v.170427, and MAFFT v7.453) (See Chapter 5). Finally, a visual interpretation of an individual MSA has been conducted to see the patterns present in each alignment, for this we used a visualization tool named Pixel Plot provided by SuiteMSA [1].

Zusammenfassung

Im Laufe der Zeit kommt es zu evolutionären Veränderungen in den Genomen der Arten. Zwei der wichtigsten Mechanismen der evolutionären Veränderungen sind Substitutionen und Indels (Insertion and Deletion). Analysen genomischer Sequenzen typischerweise auf multiple sequence alignments (MSA), die sich von Indels ableiten lassen. Die Informationen der Sequenzen, welche die Vorfahren betreffen, sind begrenzt und enthalten mehr Ereignisse wie Einfügungen, Deletionen und Substitutionen. Um die natürliche Auslese der Natur nachzuahmen, wo es kurze Auslöschungen (lange Ereignisse) gibt, werden üblicherweise simulationsbasierte Methoden verwendet, und diese natürlichen Ereignisse stellen z.B. die Einfügungen, Deletionen oder Substitutionen von genomischen Resten dar. Es gibt eine Vielzahl von Methoden und Kalkulatoren, um das MSA-Inferenzproblem zu lösen. ProPIP ist ein MSA Kalkulator, der vom Applied Computational Genomics Team (ACGT) am Institut für Angewandte Simulationen (IAS) der ZHAW entwickelt wurde. Der ProPIP stützt sich auf ein explizites Evolutionsmodell des Indels (im Gegensatz zu traditionelleren Alignern), das als Poisson-Indel-Prozess (PIP) bezeichnet wird [2]. Die vorliegende Arbeit konzentriert sich auf die Eignung des PIP (ein Ein-Zeichen-Indel-Modell), lange Indels abzuleiten, und auf die Leistung unseres Aligners ProPIP im Vergleich zu anderen traditionellen Alignern.

Die simulierten Daten bestehen aus Einzel-Indel und langen Indel-Daten. Die Eizel-Indel-Daten wurden nach dem PIP-Modell mit dem Simulator namens PIPJava simuliert, der von den Autoren von PIP, Alexandre Bouchard und Michael I. Jordan [2], zur Verfügung gestellt wurde. Und die langen Indel-Daten wurden mit dem weit verbreiteten Simulator namens INDELible v1.03 simuliert, der von William Fletcher und Ziheng Yang [6] erstellt wurde. Um reale Daten in diese Studie einzubeziehen, wird das am häufigsten untersuchte Cluster der Genfamilie γ-Proteobacteria betrachtet. Dafür werden die Proteinanordnungen verwendet, die in dem von Nancy A.Moran [13] durchgeführten Phylogenie-Rekonstruktionsexperiment verwendet wurden.

Mehrere Sequenzen sowohl aus simulierten als auch aus realen Daten wurden mit unserem Aligner ProPIP und zwei hochmodernen Alignern PRANK v.170427 (auf phylogeniebewussten Alignment-Algorithmen basierenden Aligner) und MAFFT v7.453 (ein auf Fast Fourier Transform (FFT) basierenden Aligner) rekonstruiert. Um zusätzlich die Genauigkeit unseres Aligners ProPIP zu testen, wurden die Zusatzfunktionen des Tools genutzt. Im Weiteren wird zur Abstimmung der Indel-Platzierungen eine neue Methode (k-Faktor) in ProPIP eingeführt und getestet. Nach der MSA-Rekonstruktion wird von jeder Sequenz auf jeder MSA eine Indel-Längen (siehe Abschnitt 5.2) und eine Indel-Block-Methode (siehe Abschnitt 5.3) angewendet, um die Informationen über die Lücken zu erhalten. Später wird der Unterschied in der Anzahl der Lücken zwischen der 'wahren' MSA und der abgeleiteten MSA mit der Indellängenverteilungsmethode und der Indelblockverteilungsmethode (siehe Kapitel 5) bewertet. Barüber hinaus wird der bestimmt, der vier verschiedene Qualtitätsbewertungen aus den 'wahren' MSAs (simulierte und reale) und den abgeleiteten MSAs (nach ProPIP, PRANK v.170427 und MAFFT v7.452) berechnet (siehe Kapitel 5). Schließlich wurde eine visuelle Interpretationen eines einzelnen MSAs durchgeführt, um die in jeder Ausrichtung vorhandenen Muster zu sehen. Es wurde hierfür ein Visualisierungstool namens Pixel Plot verwendet, das von SuiteMSA [1] zur Verfügung gestellt wird.

Contents

Acknowledgements i

Abstract ii

Acronyms

List of Figures

List of Tables

1 Introduction 1
- 1.1 TKF91 . . . 2

2 Background 3
- 2.1 Poisson Indel Process . . . 3
  - 2.1.1 Notation . . . 3
  - 2.1.2 Local Description . . . 3
  - 2.1.3 Global Description (Poisson Process) . . . 4

3 ProPIP 7
- 3.1 PIP Notations . . . 7
- 3.2 Discrete Gamma Distribution . . . 8
- 3.3 The k-Factor . . . 8
- 3.4 Stochastic Backtracking Algorithm . . . 9
- 3.5 Short Time Fourier Transform . . . 9
- 3.6 Prior Works . . . 10
  - 3.6.1 Simulation Settings . . . 10
  - 3.6.2 Bias Analysis of ProPIP . . . 11
  - 3.6.3 Quality check using qscore . . . 11
  - 3.6.4 Prior Results . . . 12
- 3.7 Discussion . . . 14

4 Datasets 15
- 4.1 Simulating data using PIPJava . . . 15
  - 4.1.1 Simulation Settings . . . 15
- 4.2 Simulating data using INDELible . . . 15
  - 4.2.1 Simulation Settings . . . 15
- 4.3 Real Data . . . 16

5 MSA Evaluation 17
- 5.1 MSA Reconstruction . . . 17
  - 5.1.1 MAFFT . . . 17
  - 5.1.2 PRANK . . . 17
  - 5.1.3 ProPIP . . . 17
  - 5.1.4 ProPIP under Discrete Gamma distribution . . . 18
  - 5.1.5 ProPIP under k-Factor . . . 18
  - 5.1.6 ProPIP under Discrete Gamma distribution and k-Factor . . . 19
  - 5.1.7 ProPIP under Stochastic Backtracking . . . 19

5.1.8 ProPIP under Short Time Fourier Transform . . . 19
5.2 Indel length distribution . . . 19
5.3 Indel block distribution . . . 19
5.4 Quality check using qscore . . . 20
5.5 Analysis using Pixel Plot . . . 20

6 Results and Discussions **21**
6.1 INDELible data analysis . . . 21
6.1.1 Dynamics of indels in MAFFT, PRANK and ProPIP . . . 21
6.1.2 Dynamics of indels in ProPIP under Discrete Gamma distribution . . . 22
6.1.3 Dynamics of indels in ProPIP under k-Factor . . . 25
6.1.4 Dynamics of indels in ProPIP under Discrete Gamma distribution and k-Factor 27
6.1.5 Dynamics of indels in ProPIP under Stochastic Backtracking . . . 30
6.1.6 Dynamics of indels in ProPIP under Short Time Fourier Transform . . . 32
6.1.7 Quality check using qscore . . . 34
6.2 PIP data analysis . . . 36
6.2.1 Dynamics of indels in MAFFT, PRANK and ProPIP . . . 36
6.2.2 Dynamics of indels in ProPIP under Discrete Gamma distribution . . . 37
6.2.3 Dynamics of indels in ProPIP under k-Factor . . . 39
6.2.4 Dynamics of indels in ProPIP under Discrete Gamma distribution and k-Factor 41
6.2.5 Dynamics of indels in ProPIP under Stochastic Backtracking . . . 43
6.2.6 Dynamics of indels in ProPIP under Short Time Fourier Transform . . . 45
6.2.7 Quality check using qscore . . . 47
6.3 Real data analysis . . . 50
6.3.1 Dynamics of indels in MAFFT, PRANK and ProPIP . . . 50
6.3.2 Dynamics of indels in ProPIP under Discrete Gamma distribution . . . 51
6.3.3 Dynamics of indels in ProPIP under k-Factor . . . 53
6.3.4 Dynamics of indels in ProPIP under Discrete Gamma distribution and k-Factor 55
6.3.5 Dynamics of indels in ProPIP under Short Time Fourier Transform . . . 56
6.3.6 Quality check using qscore . . . 58

7 Conclusions and Future works **61**
7.1 Conclusions . . . 61
7.2 Future works . . . 62

Bibliography

Appendix **67**

A Simulating data in INDELible **67**

B Generating guide trees for real data using PRANK **70**

C Generating parameter values given Tree and MSA **71**

D Indel length analysis in INDELible data **77**

E Indel length analysis in PIP data **81**

F Indel length analysis in real data **84**

G Pixel Plot Analysis- Additional Plots **89**

Poster

Declaration of Originality

Acronyms

ASRV Across Sites Rate Variation

CTMC Continuous Time Markov Chain

DP Dynamic Programming

FFT Fast Fourier Transform

FT Fourier Transform

INDELS Insertions and/or Deletions

ML Maximum Likelihood

MSA Multiple Sequence Alignment

NP-hard Non-deterministic polynomial-time

PIP Poisson Indel Process

SBDP Stochastic Backtracking Dynamic Programming

SP Sum-of-Pairs

STFT Short-Time Fourier Transform

List of Figures

2.1 Phylogentic tree Ω (left), and sub-tree τ_x at the given insertion position x (right), Reused from [2] 4
2.2 Global description:(A)The output of a Poisson Process on phylogeny τ, (B)Step1-Global Poisson Process, (C)Alignment and Sequence output from global process. Reused from [2] 4

3.1 Bar plots depicting the bias of ProPIP aligner in terms of MSA length 12
3.2 Box plots of MSA lengths 12
3.3 Box plots of different scores computed with Q scores for branch length 0.001 . . . 13
3.4 Box plots of Q scores for branch length 0.01 13
3.5 Box plots of Q scores for branch length 0.1 14

6.1 Pixel Plot of INDELible data- MAFFT, PRANK, ProPIP 22
6.2 Log-Log Plot of INDELible data- MAFFT, PRANK, ProPIP 23
6.3 Log-Log Plot of INDELible data- ProPIP under Discrete Gamma distribution . . . 24
6.4 Pixel Plot of INDELible data- ProPIP under Discrete Gamma distribution 25
6.5 Pixel Plot of INDELible data- ProPIP under k-Factor 27
6.6 Log-Log Plot of INDELible data- ProPIP under k-Factor 28
6.7 Pixel Plot of INDELible data- ProPIP under Discrete Gamma distribution and k-Factor 29
6.8 Log-Log Plot of INDELible data- ProPIP under Discrete Gamma distribution and k-Factor 30
6.9 Pixel Plot of INDELible data- ProPIP under Stochastic Backtracking DP (SBDP) 31
6.10 Log-Log Plot of INDELible data- ProPIP under Stochastic Backtracking DP (SBDP) 32
6.11 Pixel Plot of INDELible data- ProPIP under Short Time Fourier Transform (STFT) 33
6.12 Log-Log Plot of INDELible data- ProPIP under Short Time Fourier Transform (STFT) 33
6.13 Box plots of Q-Scores for INDELible data 34
6.14 Box plots of Modeler Scores for INDELible data 35
6.15 Pixel Plot of PIP data- MAFFT, PRANK, ProPIP 37
6.16 Log-Log Plot of PIP data- MAFFT, PRANK, ProPIP 37
6.17 Pixel Plot of PIP data- ProPIP under Discrete Gamma distribution 38
6.18 Log-Log Plot of PIP data- ProPIP under Discrete Gamma distribution 39
6.19 Pixel Plot of PIP data- ProPIP under k-Factor 40
6.20 Log-Log Plot of PIP data- ProPIP under k-Factor 41
6.21 Pixel Plot of PIP data- ProPIP under Discrete Gamma distribution and k-Factor . 42
6.22 Log-Log Plot of PIP data- ProPIP under Discrete Gamma distribution and k-Factor 43
6.23 Log-Log Plot of PIP data- ProPIP under Stochastic Backtracking DP (SBDP) . . 44
6.24 Pixel Plot of PIP data- ProPIP under Stochastic Backtracking DP (SBDP) 45
6.25 Pixel Plot of PIP data- ProPIP under Short Time Fourier Transform (STFT) . . . 46
6.26 Log-Log Plot of PIP data- ProPIP under Short Time Fourier Transform (STFT) . 46
6.27 Box plots of Q-Scores for PIP data 48
6.28 Box plots of Modeler Scores for PIP data 49
6.29 Pixel Plot of real data- MAFFT, PRANK, ProPIP 50
6.30 Log-Log Plot of real data- MAFFT, PRANK, ProPIP 51
6.31 Pixel Plot of real data- ProPIP under Discrete Gamma distribution 52
6.32 Log-Log Plot of real data- ProPIP under Discrete Gamma distribution 53
6.33 Pixel Plot of real data- ProPIP under k-Factor 54
6.34 Log-Log Plot of real data- ProPIP under k-Factor 54

6.35 Log-Log Plot of real data- ProPIP under Discrete Gamma distribution and k-Factor 55
6.36 Pixel Plot of real data- ProPIP under Discrete Gamma distribution and k-Factor . 56
6.37 Pixel Plot of real data- ProPIP under Short Time Fourier Transform (STFT) . . . 57
6.38 Log-Log Plot of PIP data- ProPIP under Short Time Fourier Transform (STFT) . 58
6.39 Box plots of Q-Scores for real data 59
6.40 Box plots of Modeler Scores for real data 60

G.1 Pixel Plot of INDELible data- ProPIP under Discrete Gamma distribution 89
G.2 Pixel Plot of INDELible data- ProPIP under k-Factor 90
G.3 Pixel Plot of INDELible data- ProPIP under Discrete Gamma distribution and k-Factor 90
G.4 Pixel Plot of INDELible data- ProPIP under Stochastic Backtracking DP (SBDP) 91
G.5 Pixel Plot of INDELible data- ProPIP under Short Time Fourier Transform (STFT) 91
G.6 Pixel Plot of PIP data- ProPIP under Discrete Gamma distribution 92
G.7 Pixel Plot of PIP data- ProPIP under k-Factor 93
G.8 Pixel Plot of PIP data- ProPIP under Discrete Gamma distribution and k-Factor . 94
G.9 Pixel Plot of PIP data- ProPIP under Stochastic Backtracking DP (SBDP) 95
G.10 Pixel Plot of PIP data- ProPIP under Short Time Fourier Transform (STFT) . . . 95
G.11 Pixel Plot of real data- ProPIP under Discrete Gamma distribution 96
G.12 Pixel Plot of real data- ProPIP under k-Factor 97
G.13 Pixel Plot of real data- ProPIP under Discrete Gamma distribution and k-Factor . 98
G.14 Pixel Plot of real data- ProPIP under Short Time Fourier Transform (STFT) . . . 99

List of Tables

6.1 Summary statistics table of INDELible data- MAFFT, PRANK, ProPIP 22
6.2 Summary statistics table of INDELible data- ProPIP under Discrete Gamma distribution 25
6.3 Summary statistics table of INDELible data- ProPIP under k-Factor 26
6.4 Indel length frequency variation in ProPIP under Discrete Gamma distribution and k-Factor 26
6.5 Summary statistics table of INDELible data- ProPIP under Discrete Gamma distribution and k-Factor 29
6.6 Summary statistics table of INDELible data- ProPIP under Stochastic Backtracking DP (SBDP) 31
6.7 Summary statistics table of INDELible data- ProPIP under Short Time Fourier Transform (STFT) 31
6.8 Summary statistics table of PIP data- MAFFT, PRANK, ProPIP 36
6.9 The indel length frequency- PIP data 36
6.10 Summary statistics table of PIP data- ProPIP under Discrete Gamma distribution 39
6.11 Summary statistics table of PIP data- ProPIP under k-Factor 41
6.12 Summary statistics table of PIP data- ProPIP under Discrete Gamma distribution and k-Factor 43
6.13 Summary statistics table of PIP data- ProPIP under Stochastic Backtracking DP (SBDP) 44
6.14 Summary statistics table of PIP data- ProPIP under Short Time Fourier Transform (STFT) 46
6.15 Summary statistics table of real data- MAFFT, PRANK, ProPIP 51
6.16 Summary statistics table of real data- ProPIP under Discrete Gamma distribution 52
6.17 Summary statistics table of real data- ProPIP under k-Factor 53
6.18 Summary statistics table of real data- ProPIP under Discrete Gamma distribution and k-Factor 56
6.19 Summary statistics table of real data- ProPIP under Short Time Fourier Transform (STFT) 57

Chapter 1

Introduction

To assess the sequence homology in species a traditionally practiced methodology is using multiple sequence alignment (MSA). The MSA help us to make a statement about homology of species and to understand how the sequence of species are related to each other. If the sequences are having a higher commonality of residues there exist a high probability of having a common ancestor to both species and both of these sequences are said to be homologous. Typically, the phylogenetic trees are used to represent the evolutionary relationship between the species, where the branch length represents the amount of change over time and the leaves/taxa represents the characters present in the column of the MSA. Furthermore, the MSA can be viewed as a table, where each row corresponds to sequence of each species and each column corresponds to the characters present at the taxa [7]. There are many software packages to do the multiple sequence alignment. When the sequences are aligned, the substitutions, insertions or deletions (indels) are the by-product of the evolutionary process. The indels/gaps position the common characters in same column to maximise the score/similarity.

Finding an optimal alignment is challenging and considered as a NP-hard problem [22]. Today there exist many tools to align multiple sequences, and given the importance of alignment step, extreme care is needed to select the alignment algorithm and its parameters. But, today, progressive methods have reduced the complexity of finding an optimal solution for multiple sequence alignment. Progressive methods align pair of sequences at a time and based on the solution to this the successive sequences are aligned. For pairwise sequence alignment there exist a scoring system which assigns a fixed score for match, and a penalty score for substitution and gap. To compute pairwise alignments given N sequences and the scoring functions, dynamic programming is widely used to obtain the optimal pairwise alignment. Dynamic programming (DP) algorithm [18] optimizes the alignment process by dividing the sophisticated larger problem into a series of smaller subproblems in a recursive manner.

To work with dynamic programming, the problem must satisfy an optimal substructure and overlapping sub-problems. The progressive DP traverses through the guide tree from top to bottom and traceback from leaf (bottom) to root (top) by running an instance of pairwise DP at each internal node. The DP uses the given scores and chooses the alignment that has the highest score. At each internal node, the progressive DP assigns an optimal sub-MSA for the corresponding subtree. The MSA at the root corresponds to the solution to the problem. An optimal solution can be achieved with time and space complexity $O(L^N)$ for a classical DP, where L is the sequence length and N is the number of sequences. Whereas, the progressive DP makes it possible to find the optimal solution in $O(m.L^2)$, where m corresponds to the number of internal nodes and L is the sequence length. It should be noted that, the progressive DP does not guarantee a global optimal solution. Because, the solutions made by the progressive DP at the internal node does not serve as the global representation of the problem. Thus, the algorithm results a solution based on the local information. Eventually, the decisions made at the internal nodes are affecting the successive nodes positively or negatively depending upon the correctness of the decision made at the internal node.

1.1 TKF91

Model based phylogenetic inference methodologies use Markov substitution models to represent sequence evolution as a stochastic process of residue substitutions along a binary tree with branch lengths representing the substitution rate [7]. Typically, the leaves of the tree are the observed sequences and the nodes are the ancestral sequences. The model parameters are usually estimated with Bayesian or Maximum Likelihood (ML) methods. These Markov models are typically assumed to have independence for sites [7].

In 1991, Thorne, Kishino and Felsentein [20] worked on the DP algorithm using a time-reversible Markov model. This work is then termed as TKF91. In this classical TKF91 model, the substitutions are considered as a continuous-time Markov chain (CTMC) processes and indels as a continuous-time birth-death process. Moreover, CTMC allows us to compute the joint probability of a given tree and alignment. In TKF91, the evolutionary processes are modeled along the sequence using a set of exponential random variables (3n+1 independent exponential random variables) for every mutation event (substitution, insertion and deletion). These are drawn with exponential rates: λ_{TKF} represents an insertion event and μ_{TKF} a deletion event. When substitution occurs, variables are taken from a substitution rate matrix Q with parameters θ= (λ and μ) to choose a new residue. The model decides on the events based on the smallest exponential random variables (RV). The value of smallest RV decides the next event. And, the computational time complexity of calculating the marginal likelihood (sum over all likelihoods of all events) grows exponentially with the number of sequences O(L^N).

This report is further organised as six Chapters, list of literature and finally an appendix which provides the Matlab scripts used and additional plots produced using the methods described in this report. The Chapter 2 provides the background details about single character indel model, PIP [2]. Then, the Chapter 3 explains our progressive aligner named ProPIP, its features and the previous works that were conducted to understand bias in ProPIP. In Chapter 4, the datasets used in this study are briefly explained. The Fifth Chapter is devoted to the MSA evaluation methods that are used in this study and, the purpose of Chapter 6 is to select and analyse each evaluation methods explained in Chapter 5 and provide statistical results of these methods with a short discussion. Finally, the Chapter 7 concludes the report by answering the objectives of this thesis along with suggestions for the future works.

Chapter 2

Background

In this Chapter the theory and mathematical formulation of PIP is outlined. PIP is a single character indel model, which provides a model based approach for joint probabilistic inference of trees and alignments, developed as the successor to the classical TKF91 model in order to improve the computational complexity. The marginal likelihood under TKF91 model require a computational complexity of $O(L^N)$, where L is the length of the sequence and N is the number of taxa. However, PIP improves the computational complexity to $O(|L|.|m|)$, where $|L|$ is the number of observed taxa and $|m|$ is the number of columns present in the alignment [2].

2.1 Poisson Indel Process

Poisson Indel Process (PIP) is a string-valued evolutionary model constructed by Bouchard and Jordan [2] as the successor of TKF91 [20]. To begin with, PIP has two descriptions: a local description, which is same as the local continuous time Markov process as described in TKF91 [20] and the second, a global description as a Poisson process.

2.1.1 Notation

In this section the notations used in PIP are explained. The phylogeny is depicted as a continuous set of points and represented as τ. A phylogenetic tree is represented as $\tau = (\mathbf{V}, \epsilon, \mathbf{b})$ with $\mathbf{N}$ leaves, where the ϵ denotes the set of edges, $\mathbf{V}$ represents the set of branching points and $\mathbf{b}$ as the branch length. The branch length of a node $\mathbf{b}(v)$, where $v \in \mathbf{V}$, is the length of the edge from the parent node $\text{pa}(v)$ to root v. Leaves $L \subset \mathbf{V}$ denotes $\mathbf{N}$ observed taxa, represented by strings of characters from a finite set of alphabets $\sum$ (nucleotides, amino acids or codons). The root of the phylogenetic tree, Ω is the most recent common ancestor of all leaves. A sub-tree of τ rooted at a point $\mathbf{x}$ in the phylogenetic tree will be represented by τ_x. The sub-tree is considered by excluding all the nodes in the phylogenetic tree τ that are not descendants of $\mathbf{x}$. As an example, in the figure 2.1, a rooted phylogenetic tree with a common ancestor Ω and four leaves is shown, the leaves/taxa contains *M.Sylvanus*, *M.Fuscata*, *Hylobates* and *H.Sapiens*. The point marked with a star is considered as an insertion point $\mathbf{x}$. And, at the right most figure is the rooted sub-tree τ_x.

2.1.2 Local Description

PIP follow the same structure of stochastic process used in TKF91. When TKF91 utilizes $3n+1$ exponential RVs to predict the next event (n for substitutions, $n+1$ for insertions and n for deletions), the PIP uses a different approach with $2n+1$ exponential RVs (n RVs for substitutions, 1 global RV for insertions with rate λ and n RVs for deletions, each of rate μ). To find the next event and its waiting time a general algorithm is to simulate one exponential RV for each of the events (substitution, insertion and deletion) and on each of the location where these events can be applied. The smallest of the independent RVs determines the nature of the next mutation and also the time at which it occurs.

When an insertion occurs, its position is selected uniformly at random over the sequence. In detail, if there exist a real number r_i between the interval [0,1] assigned to each character in the string/sequence in increasing order: $0<r_1<r_2<....<r_n<1$. Then, when an insertion happens, a new

real number r' is sampled uniformly in the interval [0,1] and this new character is inserted to the sequence at the unique position (with a probability 1) without disturbing the order of the sequence of real numbers [2].

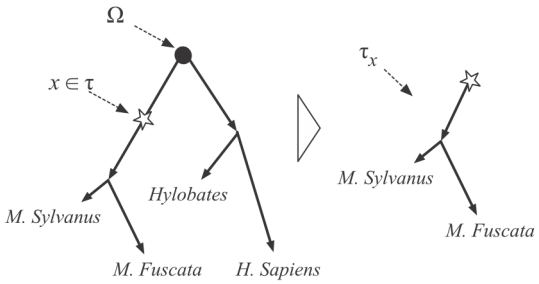


Figure 2.1: Phylogentic tree Ω (left), and sub-tree τ_x at the given insertion position x (right), Reused from [2]

2.1.3 Global Description (Poisson Process)

The second description of PIP model represents a global Poisson process. This process includes two steps. The first step consider insertion events and the second step consider deletion and substitution events. In the first step, a set of insertion points are sampled from a Poisson process defined on the phylogeny τ. For instance, the stars in the figure 2.1. The random set $\prod = \{X_i\}$ represents the location of the insertion points. To elaborate, the atomic insertions are depicted as Poisson events with rate measure $v(dt) = \lambda(\tau(dt) + \mu^{-1}\ \delta_\Omega(dt))$, where λ is the insertion rate, μ is the deletion rate and $\delta_\Omega(.)$ is called the Dirac's delta function. This formula keeps the equilibrium of the expected sequence length during the complete evolutionary process [2].

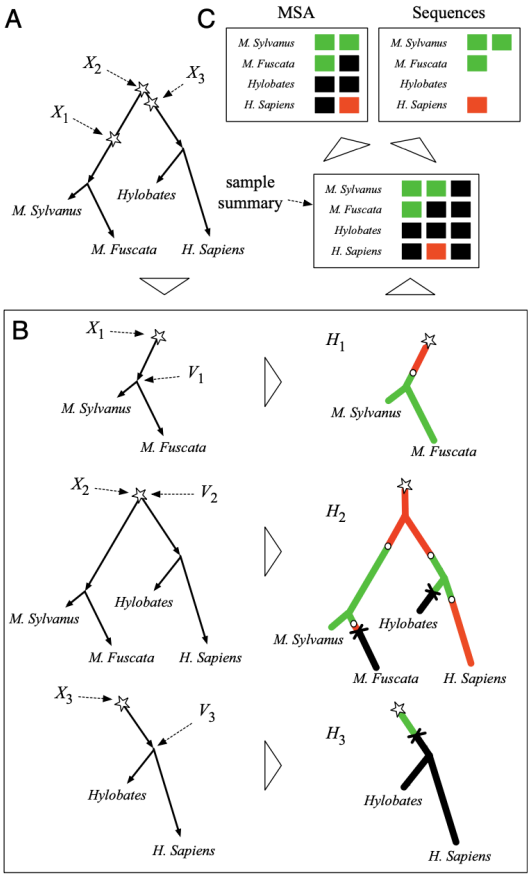


Figure 2.2: Global description:(A)The output of a Poisson Process on phylogeny τ, (B)Step1-Global Poisson Process, (C)Alignment and Sequence output from global process. Reused from [2]

The second step consider each insertion locations $\{X_i\}$ and visit them one by one randomly. Each visit contain two steps, the first step (See figure 2.2.A) uses the sample from the Poisson process on phylogeny τ. In this, the sub-trees τ_x, are extracted from the given rooted phylogenetic tree, Ω. In the second step the other mutations (substitution and deletions) are exchangeable continuous time Markov chains. This step simulate the sub-tree τ_x and predict the evolution of the inserted character. To do this, the state space for the CTMC is extended with a deletion symbol ϵ, $\sum_\epsilon = \sum \cup \epsilon$. The CTMC generates an homology path, H_i which is a random map of all the points in the phylogeny τ to $\sum_\epsilon$ [2]. Note that, the branches in black color in the figure 2.2.B represents the characters deleted.

At each internal node v the marginal likelihood is computed. For this the homology paths $\{X_i\}$ are conditioned. The prior insertion probability $i(v)$ to insert a character on a branch between $pa(v)$ and v, proportional to $b(v)$ is defined as [2]:

$$i(v) = \begin{cases} \frac{b(v)}{\|\tau\|+\mu^{-1}} & , v \neq \Omega \\ \frac{\mu^{-1}}{\|\tau\|+\mu^{-1}} & , v = \Omega \end{cases} \tag{2.1}$$

Note that, the atomic root mass point probability ($i(v)$ when $v = \Omega$) in the above equation satisfies the probability condition $\sum_{n=1} i(v) = 1$ and $\|\tau\|$ is the total branch length in the topology. The survival probability $\beta(v)$ of the character inserted on a branch between $pa(v)$ and v is given by [2]:

$$\beta(v) = \begin{cases} \frac{1-exp(-\mu b(v))}{\mu b(v)} & , v \neq \Omega \\ 1 & , v = \Omega \end{cases} \tag{2.2}$$

The marginal likelihood of MSA, of length $|m|$ and number of taxa N is defined as

$$p_\tau(m) = \mathbb{E}[\mathbb{P}(M = m||X|)] \tag{2.3}$$

The above formula can be rewritten as,

$$p_\tau(m) = \sum_{n=|m|}^{\infty} \mathbb{P}(|X| = n).\binom{n}{|m|}.(p(c_\phi))^{n-|m|} \prod_{c \in m} p(c) \tag{2.4}$$

The first term in Eq.2.4 is to calculate the probability of sampling n homology paths, the second term, is simply the number of ways to choose $|m|$ insertion events from the overall insertion events, in the last term $p(c)$ is the likelihood of a single column c and $p(c_\phi)$ is the likelihood of a single column with complete gap. Representing the above equation using $\varphi(.)$ function, where $p(c_\phi) \in$ (0,1) and $|m| \in$ {1,2,3,.....} gives:

$$\varphi(p(c_\phi),|m|) = \frac{\|v\|^{|m|}\, exp(\|v\|\,(p(c_\phi) - 1))}{|m|!} \tag{2.5}$$

where $\|v\|$ is the normalizing Poisson intensity, $\|v\| = \lambda(\|\tau\| + \frac{1}{\mu})$. For each column in C, the column probability can be computed using:

$$\begin{aligned} \mathbb{P}(C = c) &= \sum_{v=V} \mathbb{P}(V = v)\mathbb{P}(C = c|V = v) \\ &= \sum_{v=V} \mathbb{P}(V = v) f_v \end{aligned} \tag{2.6}$$

where $\mathbb{P}(V = v)$ is same as $i(v)$ and f_v is a slightly revised standard Felsenstein peeling recursion (dynamic programming) [4] output on the sub-tree τ_v, which computes the probability of the substitution-deletion process of a single character. The SI Appendix, Section 2 of [2] provides a detailed mathematical derivation of f_v. The computation of Felsenstein peeling recursion for one column takes a linear time complexity $O(N)$, therefore the total computation can be computed in $O(N.|m|)$, where N is the observed taxa and $|m|$ is the number of columns in the alignment.

Chapter 3

ProPIP

A new progressive Dynamic Programming is introduced under the PIP evolutionary model, which aligns N sequences ($N > 2$) in polynomial time using maximum likelihood (ML) approach. The ACGT team from IAS, ZHAW have implemented (in C++ by Maiolo et al.[15]) and made available a frequentist progressive multiple sequence aligner under explicit evolutionary models of substitutions, insertions and deletions named ProPIP [15] [12]. This chapter briefly explains the theory behind ProPIP, features implemented in ProPIP and the previous studies conducted on ProPIP.

While PIP outlines an evolutionary process mathematically, a computational implementation of a multiple sequence aligner was developed by Maiolo et al.[15] that uses PIP and a DP algorithm. The algorithm starts with a pairwise alignment of sequences (sequence-1 and sequence-2) at the leaves and traverses along a guide tree in post-order until the root Ω is reached. The marginal likelihood at each internal node is calculated as three scores (match score for a match of sequences/sub-MSA's, one gap score for a sequence/sub-MSA-1 aligned to a gap and one gap score for a gap aligned to a sequence/sub-MSA-2). At each internal node v, the algorithm maximize the likelihood while aligning the sub-MSA's on the children nodes under the PIP model. The marginal likelihood of the completely gapped columns (Eq. 2.5) is non-monotonic in alignment length therefore an additional dimension is required [15]. Thus, our algorithm uses four 3D matrices *match* S^M, *gapX* S^X and *gapY* S^Y in forward phase and a trace-back matrix S^T in backward phase. In each of the matrices the 1^{st} dimension represents the length of sequence-1, 2^{nd} dimension corresponds to length of sequence-2 and 3^{rd} dimension is used to represent the sum of the size of the sequence-1 and sequence-2. During the computation, the algorithm considers one layer at a time along with the 3D matrices and fills the matrices with likelihood values by keeping the trace in the trace-back matrix S^T, that is, the name of the DP matrix with highest likelihood at the same position. If the maximum likelihood is not unique a random choice is made. An optimal alignment is determined by backtracking along the trace-back matrix S^T. Depending on the value stored in the S^T matrix the corresponding columns are aligned in one of the three states (*match*, *gapX* or *gapY*) thus generating the alignment with the highest likelihood under PIP. This approach gives an overall complexity of $O(N.L^3)$, where N is the column length and L is the maximum sequence length. However, by parallelizing the computation with high speed processors the computational complexity can be reduced to $O(N.L)$, where N is the number of sequences and L is the length of the longest sequence.

3.1 PIP Notations

Other than the mathematical formulation of PIP, Bouchard and Jordan [2] created a JavaScript, to simulate the data (sequences and True MSA's) depending on defined parameters, called PIPJava. The asymptotic expected sequence length, η is defined as the ratio of λ and μ. And, the insertion-deletion (indel) intensity, ζ is defined as the product of λ and μ.

$$\eta = \lambda/\mu \tag{3.1}$$

$$\zeta = \lambda * \mu \tag{3.2}$$

From the above equations, the insertion rate (λ) and deletion rate (μ) can be derived.

$$\lambda = \eta * \sqrt{\zeta/\eta} \tag{3.3}$$

and,

$$\mu = \sqrt{\zeta/\eta} \tag{3.4}$$

3.2 Discrete Gamma Distribution

Many features have been implemented in this tool. One of them is the Discrete Gamma distribution. To tune different regions within the sequence, thereby minimizing the substitution rate variation among sites (ASRV) [23], Discrete Gamma distribution is utilized. In detail, the discrete gamma distribution consider different categories within the sequence and assumes that each site experience a substitution rate r, which is an i.i.d. random number according to given distribution $g(r)$. Moreover, by assuming that k number of substitutions are possible in a site during an evolutionary interval of time t, where the substitution events are assumed as a Markov process of substitutions or a Poisson process at a given rate r, the distribution of k can be written as:

$$f(k) = \int_0^\infty \frac{(r.t)^k e^{-r.t}}{k!} g(r) dr \tag{3.5}$$

To compute $\mathbb{E}(k)$ and $Var(k)$ the above equation 3.5 is used, where $\mathbb{E}(k)$ as the first moment of the distribution $f(k)$ and $Var(k)$ as the second moment of $f(k)$.

$$\mathbb{E}(k) = \sum_{k=0}^{\infty} (k.\, f(k)) \tag{3.6}$$

$$Var(k) = \sum_{k=0}^{\infty} (k^2.\, f(k)) - (\mathbb{E}(k))^2 \tag{3.7}$$

By substituting Eq.3.5 in Eq.3.6 gives $\mathbb{E}(k) = \mathbb{E}(r).t$ and substituting Eq.3.5 in Eq.3.7 gives $Var(k) = Var(r).\, t^2 + \overline{r}.\, t$, where $\overline{r} = \mathbb{E}(r)$. From this final equations of $\mathbb{E}(k)$ and $Var(k)$ the equation for expected value of r, $\mathbb{E}(r)$ can be represented in terms of $\mathbb{E}(k)$ and the equation for Variance of r, $Var(r)$ in terms of $\mathbb{E}(k)$ can be reformed as:

$$\overline{r} = \mathbb{E}(r) = \frac{\mathbb{E}(k)}{t} \quad \text{and} \quad Var(r) = \frac{Var(k) - \mathbb{E}(k)}{t^2} \tag{3.8}$$

The probability density function (PDF) of discrete gamma distribution is given by

$$g(r) = \frac{(\frac{\alpha}{\overline{r}})^\alpha}{\Gamma(\alpha)} \,.\, r^{\alpha-1} \,.\, e^{\frac{-\alpha.r}{\overline{r}}} \tag{3.9}$$

where α is the shape parameter and $\Gamma(\alpha)$ is the gamma function, for all positive integers $\Gamma(\alpha) = (\alpha - 1)!$.

To apply gamma distribution into our tool the mathematical equations used in PIP are reframed. The detailed mathematical derivation of PIP equations under gamma distribution can be viewed in Section 4.2 of [16]. Note that, when we use relatively smaller α value there will be a high among sites rate variation, and when we use relatively higher α value then there will be a low degree of variation.

3.3 The k-Factor

A new parameter called k-factor or 'k' is introduced into our tool ProPIP through this research work. The "k" in the k-Factor represents a continuous value greater than 0. Choosing an optimal value for k is challenging and an ongoing research, which needs to predict the MSA length, the difference between true and inferred indel lengths and other factors depending on k. The k-Factor or k-Scale (used in Plots) is applied to the basic PIP equations explained in the above section 3.1.

When both insertion and deletion rates are scaled with k-Factor or 'k' then the new scaled version of λ and μ can be represented as:

$$\lambda' = k.\lambda \quad \text{and} \quad \mu' = k.\mu \tag{3.10}$$

and, By Substituting 3.10 in 3.3 and 3.4 the basic PIP equations can be modified as:

$$\eta' = \frac{\lambda'}{\mu'} = \frac{k.\lambda}{k.\mu} = \frac{\lambda}{\mu} \tag{3.11}$$

and,

$$\zeta' = \lambda' * \mu' = k.\lambda * k.\mu = k^2.\lambda.\mu \tag{3.12}$$

The equation 3.11 clearly says that, scaling λ and μ with k-factor is not affecting the sequence length. However, from equation 3.12, it is seen that the value of the parameter k plays a key role to adjust the indel intensity. The role of this new method, k-factor in indel dynamics is tested through this research work.

3.4 Stochastic Backtracking Algorithm

From the prior work results [Ref: Section 3.6.4] it has been shown that the greedy nature of progressive DP introduces negative biases. Inspired from the Stochastic backtracking algorithm of Mückstein et al. [17], Maiolo et al. [15] have implemented a Stochastic backtracking algorithm using DP (SBDP) under PIP. This section briefly explains the basic concepts and theory about the same. The detailed explanation of the mathematical formulations and other notations are provided in Chapter 3 of [16].

In ProPIP the DP algorithm follows a progressive approach. Initially, a pair of sequences at the leaves are considered and the algorithm traverses along a guide tree in post-order until the root, Ω. At each internal node the marginal likelihood is calculated, and the alignment is decided based on the maximum likelihood value. However, the progressive approach neglects the possibility of having multiple optimal solutions and make decisions based on a single sub-optimal solution. Therefore, to avoid this error the study by Mückstein [17] is referred and implemented (by Maiolo et al. [15]) as a new Stochastic Backtracking algorithm using DP under PIP and made available as a supplementary feature to our tool ProPIP.

A theoretical explanation about SBDP is explained, however, finding the optimal T is an ongoing research. The Stochastic Backtracking algorithm using DP (SBDP) under PIP is driven by a distortion/temperature parameter, T. In short, the distortion parameter is the key to this method. In contrast to the classical DP algorithm, the SBDP doesn't store any information regarding the maximum likelihood or in other words it consider all the sub-optimal solutions obtained during the progressive DP algorithm at each internal nodes. The distortion parameter controls the stochastic path taken during the backtracking phase. For instance, using this algorithm when T= 0 returns the optimal alignment (the paths taken will be closer to diagonal and at lower T values the algorithm assigns a probability close to 1), whereas when T= ∞ the number of traceback paths will be very large, which means that all sub-optimal sub-MSA's have equal probability to get selected. Therefore, the solution at T= ∞ will be purely random. To note, there exist no traceback matrix in SB algorithm as in the classic DP under PIP.

3.5 Short Time Fourier Transform

To improve the computational complexity of our tool Maiolo et al.[15] implemented a multi-scale Short Time Fourier Transform. As explained earlier in this Chapter our progressive DP algorithm aligns two MSA's under the PIP model by maximum likelihood method. This algorithm requires a computational complexity cubic in the sequence length $O(N.L^3)$ as it is designed in 3D DP matrices. To speed up, as a solution to this, a possible strategy is to predict the candidate homologous segments prior to the alignment process, for the purpose of filtering out the non-promising regions in the DP matrix [16]. The traditional aligner MAFFT relies on a Fast Fourier Transform (FFT) to do the same. Inspired from this Maiolo et al. [16] implemented a new STFT-based approach to identify the homologous regions prior to the analysis and applied this along with

the progressive DP-PIP framework. This section explains briefly the idea behind STFT feature implemented in ProPIP. And, more mathematical explanations are explained in Chapter 5 of [16].

The idea to reduce the computational complexity is that, instead of aligning two complete sequences, we split the sequences in portion and align only a portion of it and then merge these sub alignments (as the computational time complexity is dependant on the sequence length). Through this way the size in memory and number of computations can be decreased, thereby reducing the computational complexity of the whole process. One of the methods to identify the homologous regions in the input multiple sequences is using the technique $cross - correlation$ approach. Cross-correlation is a tool that identify the peaks/homologous regions occurring in the two input functions/sequences. In other words we can say that the cross correlation is measuring the shape of the function, if both sequences are having higher values (the conversion of amino acid sequences is provided in [16]) then they are said to be highly cross correlated. The cross correlation of two sequences $S1$ and $S2$

$$f_k \equiv f[k] = \sum_{1 \leq i, i+k \leq L} \hat{s}_{i,1} \cdot \hat{s}_{i+k,2} \tag{3.13}$$

is computed as a function of the lag k between the sequences, where the subscript 1 and subscript 2 represents the sequences, i and $i + k$ denotes the column indices in the signals $\hat{s}_1$ and $\hat{s}_2$ respectively. To note, when the sequences $S1$ and $S2$ are not same then $L = max(L1, L2)$ and further applied zero padding method with pad size as the absolute length difference of the two sequences. The cross correlation is same as taking convolution of both sequences. The convolution theorem allow us to compute the above cross-correlation equation 3.13 by means of Fourier Transform [9]. Also, the Fourier transform of a convolution of two signals is equivalent to the point wise product of their Fourier transforms. Under FT it is possible to accelerate the computation using Fast Fourier Transform (FFT) which reduces the computational time from $O(L^2)$ to $O(L.logL)$, where L is the average sequence length. However, a disadvantage of using the FT approach is that it only provides information in pure frequency domain and not in time domain. In detail, to identify the homologous regions FT can be applied but information regarding the location and lag of the patterns occurring in the sequence is not provided by FT. For instance, MAFFT address this issue by a column scoring technique (See Katoh et al 2002 [9]). But our algorithm uses a more advanced method that is using a Time-Frequency transformation using a window function, Maiolo et al. refer this as a multi-resolution short-time Fourier Transform (STFT), which provides both time and frequency information about the homologous blocks in the functions/sequences simultaneously. Applying STFT provides the information regarding position, time, and duration or length of the pattern. The basic requirements for STFT based simulation implemented in our aligner ProPIP are a window function (FIR filters: welch, hamming, hanning or Bartlet) with it's window size (32, 64 or 128), a guide tree, input multiple sequence and ProPIP parameters (insertion rate λ and deletion rate μ).

3.6 Prior Works

In the prior work, an analysis of bias on MSA's inferred by progressive PIP aligner (ProPIP) [12] were conducted based on a simulation based approach. In the prior study, we (Jithin Mathew Peechatt, Eldhose Poulose and Cam Phuong Ta) simulated data using PIPJava [2] built under an explicit evolutionary model called Poisson Indel Process (PIP) (Section 2.1). For simulation we used different parameter settings as explained below.

3.6.1 Simulation Settings

- Simulation residues: Nucleotides
- Tree topology: i) perfectly balanced and ii) ladder topology
- Number of Taxa/leaves: 8
- Branch length: fixed branch lengths. 0.1, 0.01, 0.001
- Substitution model: K80, where $k = 4.0$
- Asymptotic expected sequence length (η): 1000 nucleotide residues
- Indel intensity (ζ): 10, 100, 200

- Replicates: 100 for each topology-intensity pairs within each branch length.

The simulated data consist of 600 (2 topologies * 3 intensities * 100 replicas) sequences and MSA's. The simulated sequences were provided as an input to ProPIP along with other parameters of ProPIP such as; (tree τ (tree topology and branch length), insertion rate (λ), deletion rate (μ) and the substitution model, Q). All parameters were fixed to 'true' parameters as used in the data simulation using PIPJava (Section 3.6.1). Note that, the insertion rate (λ), and deletion rate (μ) were calculated using the Eq.3.3 and Eq.3.4.

3.6.2 Bias Analysis of ProPIP

To assess the performance of our progressive aligner ProPIP [12], as a prior work, we analyzed the inferred MSA's using different methods. Since the prior study was focused on the bias analysis of ProPIP we conducted a comparative study on 'true' vs. Inferred MSA's in terms of the MSA length and a quality check using the classical qscore software [19].

To begin with, the bias of an estimator [11] is defined as the difference of its expected value and the observed value, where the value in our study represents the size of the MSA. When an estimator outputs zero bias this tool is said to be unbiased and if the value is negative it is said to be negatively biased else positively biased. Mathematically, bias can be written as:

$$Bias[\hat{\theta}] = \mathbb{E}[\hat{\theta} - \theta] \tag{3.14}$$

In terms of MSA length, the eq.3.14 can be rewritten as:

$$Bias[\text{Inferred MSA length}] = \mathbb{E}[\text{Inferred MSA length-True MSA length}] = \sum_{i=1}^{n} \frac{\hat{l_i} - l_i}{n} \tag{3.15}$$

where n is the number of replicates and $\hat{l_i}, l_i$ represents the inferred and true MSA lengths respectively. Since we need to compute the variation of MSA lengths the relative bias is calculated, which is defined as the ratio of the difference of inferred and 'true' MSA lengths to 'true' MSA length.

$$Relativebias[\text{Inferred MSA length}] = \mathbb{E}[\frac{\text{Inferred MSA length-True MSA length}}{\text{True MSA length}}] \tag{3.16}$$

3.6.3 Quality check using qscore

To analyse the quality of the alignments inferred by ProPIP, four different standard quality scores were computed using the quality scoring software named qscore [19]. qscore compares two multiple sequence alignments (the "observed" alignment and the "reference" alignment) and generate four different scores. From the *usage.cpp* script in qscore source file [19], the definitions for the scores are as follows:

- The PREFAB Q score (aka the Balibase sum-of-pair score):

$$\sum_{i=1}^{N} \frac{correctly\ aligned\ pairs}{aligned\ pairs\ in\ reference} \tag{3.17}$$

  where i is i^{th} column in MSA's and N is length of the MSA's

- The Modeler score:

$$\sum_{i=1}^{N} \frac{correctly\ aligned\ pairs}{aligned\ pairs\ in\ observed} \tag{3.18}$$

- The Balibase TC (total column) score:

$$\sum_{i=1}^{N} \frac{correctly\ aligned\ columns}{aligned\ columns\ in\ reference} \tag{3.19}$$

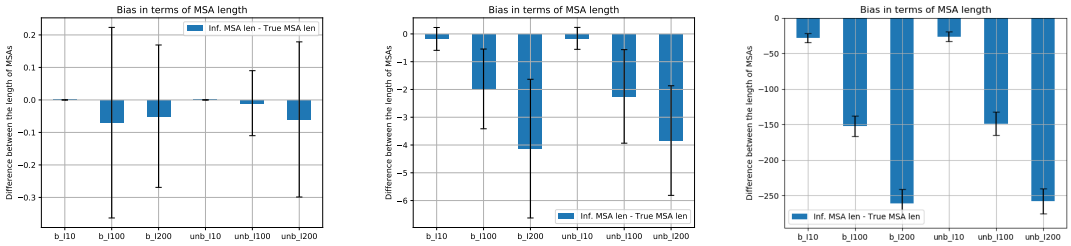
- The Shift score:

  If the two alignments are identical the shift score is 1. If a character is aligned to c_1 in alignment-1 (observed MSA), and character c_2 in alignment-2 (reference MSA), the shift score is the number of residues between c1 and c2. The mathematical formulations are discussed in detail in [21]. The shift score [21] assesses the misalignment between two alignments.

3.6.4 Prior Results

The prior results related to bias of our tool showed that, the bias grows negatively as the branch length or intensity increases in almost all the topology-intensity combinations. Especially, within the same branch length, these differences become larger as the intensity rises. In the plots given in this section, the topology-intensity combinations denoted in the x-axis explains the following:

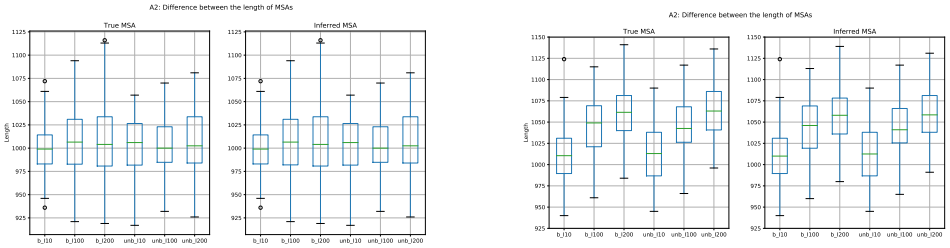
- b_I10: balanced tree topology, indel intensity 10
- b_I100: balanced tree topology, indel intensity 10
- b_I200: balanced tree topology, indel intensity 200
- unb_I10: unbalanced/ladder tree topology, indel intensity 10
- unb_I100: unbalanced/ladder tree topology, indel intensity 100
- unb_I200: unbalanced/ladder tree topology, indel intensity 200



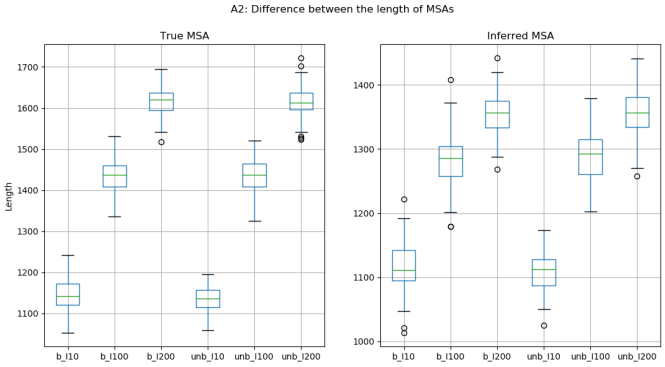
(a) branch length 0.001 (b) branch length 0.01 (c) branch length 0.1

Figure 3.1: Bar plots depicting the bias of ProPIP aligner in terms of MSA length

Note that, the average difference between 'true' and inferred MSA's (over 100 replicates in each topology-intensity combination) in length is shown along with its standard deviation as error bars in Figure 3.2.



(a) branch length 0.001 (b) branch length 0.01



(c) branch length 0.1

Figure 3.2: Box plots of MSA lengths

The Figure 3.1 shows the bias analysis results. Except for the case of balanced tree topology in branch length 0.001 [fig:3.1a], bias grows in the negative direction (showing underestimation of events by ProPIP) as intensity increases in both topologies as shown in Figure 3.1. Also, in Figure 3.2 , the box plots, which contains the standard deviation of the quantity over 100 replicates, for the

parameter I=10 in (c) and in all cases of (a) and (b) overlaps each other, suggesting non-significant differences between true and inferred MSA's in the quantity being analyzed.

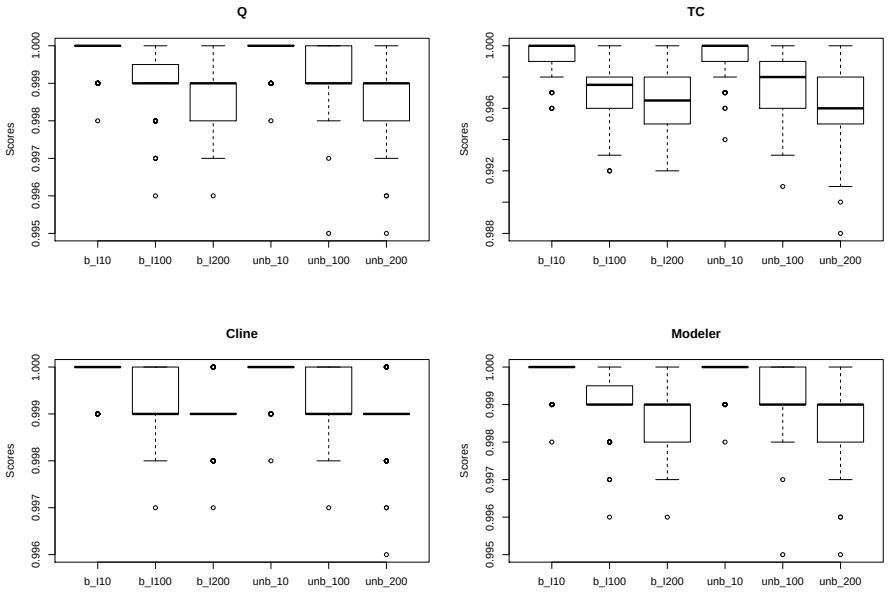


Figure 3.3: Box plots of different scores computed with Q scores for branch length 0.001

Among the quality scores used, the only difference in Eq. 3.17 and Eq. 3.18 is in the denominator, where in Q score equation the denominator is the pairs in the reference MSA and in Modeler score equation the denominator is the pairs in the test MSA. Again for all topology-intensity combinations the box plots corresponds to all scores are plotted. Since Q and Modeler results graph are identical in the Figure 3.3 it can be derived that the aligned pairs are identical between reference and test MSA's. Also, it can be observed from the Figures 3.3, 3.4 and 3.5 that as the branch length or intensity increases, the scores decrease consistently in both balanced and unbalanced tree topology. This trend is observed in all the scores produced by qscore. It is interesting to note that for branch length 0.001 and intensity 10, almost all replicates scored 1 in all the four different scores calculated.

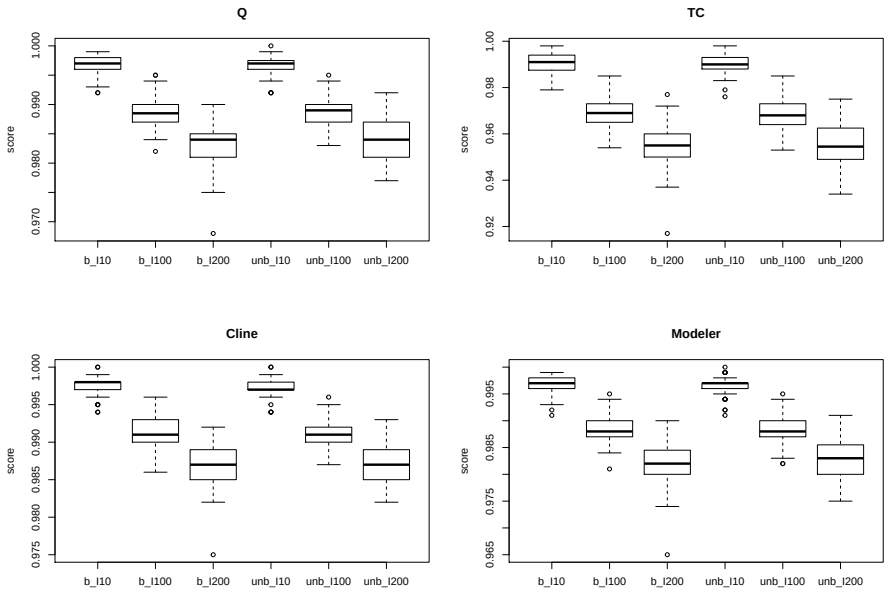


Figure 3.4: Box plots of Q scores for branch length 0.01

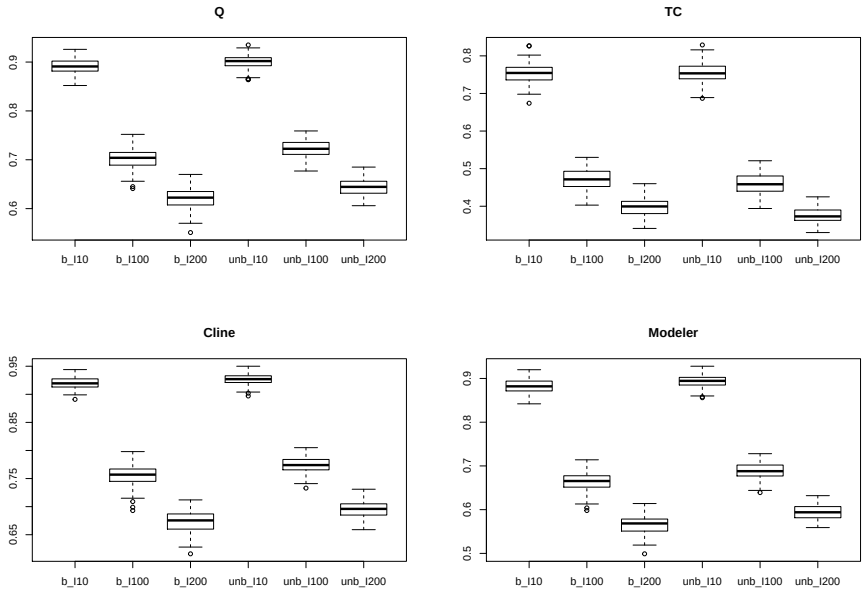


Figure 3.5: Box plots of Q scores for branch length 0.1

3.7 Discussion

The bias analysis concluded that there is a negative bias in our aligner (ProPIP) as the branch length or indel intensity increases. This could be an effect of the greedy nature of the progressive PIP algorithm. From equations 3.3 and 3.4, it is evident that intensity is directly proportional to insertion rate λ and deletion rate μ. Hence as branch length (substitution rate) and intensity increases, the number of mutation events/changes along the evolutionary history also rises and the greedy and progressive nature of this algorithm could be overlooking some of those events. This could explain the observations/results in our prior work. The prior work put forward a suggestion to check for patterns in indel distribution that could be causing these observations with varied parameters and other features implemented in the aligner. The MSA evaluation methods [See Chapter 5] used in this study are based on these prior work results.

Chapter 4

Datasets

In this chapter, the datasets used for this study is explained. Simulated data and real datasets are considered for the study. Many tools are available to simulate the evolutionary sequence data. PIPJava is a simulator written in Javascript provided by Bouchard and Jordan [2] to simulate data under Poisson Indel Process or PIP model. Next, a long indel simulation software named INDELible is included, provided by Fletcher and Yang [6], which allows the user to simulate data under multitude of indel-length distributions. Finally, protein alignments from Nancy A. Moran [13], that they used in the reconstruction of organismal phylogeny study are used for this research work.

4.1 Simulating data using PIPJava

4.1.1 Simulation Settings

We simulated the data (sequences, true alignments and trees) under PIP model using the PIPJava simulator using the following parameters:

- Simulation residues: Amino Acid
- Tree topology: Balanced, Symmetric tree with exponential branch lengths with tree rate 1/8
- Number of Taxa/leaves: 8
- Substitution model: WAG
- Asymptotic expected sequence length (η): 200 aminoacid residues
- Indel intensity (ζ): 0.5
- Insertion rate (λ):10
- Deletion rate (μ):0.05
- Replicates: 100

4.2 Simulating data using INDELible

The INDELible is a software that allow us to simulate the molecular sequences with varying indel length using different parameters. INDELible v1.03 allows several options to the user to simulate data. More frameworks in INDELible v1.03 can be viewed in [6]. In this study, however, the following configuration is used:

4.2.1 Simulation Settings

- The data (amino acid sequences and true MSAs) is simulated using the same tree that is used in the simulation of PIP data (Section 4.1).
- Tree Topology: Balanced, Symmetric tree with exponential branch lengths with tree rate 1/8.

- The data is simulated using the WAG substitution matrix with initial 200 amino acid residues at the root.
- The indel model is set to Power law distribution with parameters $a = 1.7$ and $m = 20$. In detail, the value of a is a decimal ($a > 1$) and m is an integer ($m > 1$), which represents the maximum indel length. Moreover, many studies has been conducted in the past about the indel length distribution. Empirical studies by Gonnet [8] suggest to follow a Zipfian (power law) distribution and the value for the parameter a best fit the data when the value of a ranges between 1.5 and 2. Therefore, we selected the value of a as a= 1.7. The value of m determines the maximum indel length at the root and this is set randomly to 20.
- The indel rate is fixed to 0.05 (same deletion rate used in PIP simulation). In order to keep the insertion and deletion rate in an equilibrium state so that the MSA doesn't shrink or grow rapidly.
- To note, by setting the indel rate parameter value the simulation will run under same insertion and deletion rates. However, it is also possible to set different rates and different models for insertion and deletion events.
- The number of replicates simulated is 100.

4.3 Real Data

To understand the indel dynamics in real data (protein alignments) this study used the 13 bacterial sequence alignments provided by Emmanuelle Lerat, Vincent Daubin and Nancy A. Moran, which they inferred using CLUSTALW 1.8 for their study [13] about the reconstruction of organismal phylogeny. An initial analysis is carried out with four protein alignments (Prot_atpB, Prot_bioB, Prot_gyrA and Prot_rpoA) out of 205 protein alignments mentioned in their paper [13]. Even though these protein alignments were generated using CLUSTALW 1.8 program [3] this study consider the MSAs from PRANK v.170427 as the 'true'/reference alignment due to its exceptionally accurate results in previous studies [15] and also due to insufficient information about these protein alignments. Moreover, the tree topologies corresponds to these 4 protein alignments with 13 taxa is inferred using PRANK v.170427 using its $-treeonly$ option. More details of other protein alignments can be viewed in [5].

Chapter 5

MSA Evaluation

This Chapter briefly explains the basic configuration settings used to generate MSA using different tools in Section 5.1 and the research methodologies applied to do the MSA evaluation in Section 5.2 to 5.5. As explained in the Chapter 3, this study uses simulated as well as real data. For all data types the MSAs are reconstructed using MAFFT v7.453 [10], PRANK v.170427 [14], ProPIP [12] and ProPIP under its several features. Once the data is generated the study analyses the MSAs using different methods. First, an initial study aimed at identifying the amount of the indels present in each sequence for each reference MSA and observed MSA is conducted (referred to as Indel length distribution, Section 5.2). Next, a method to understand the effect of removing the misplaced single characters in between the gaps to inferred indel length distribution, called Indel block method (Section 5.3) is experimented on all inferred MSAs. To further determine the quality of the inferred MSAs the traditional quality scoring program, qscore [19] is utilized. Finally, to visualize and differentiate the indel patterns in the reference MSA and the observed MSA the study make use of the Pixel Plot tool provided by SuiteMSA [1].

5.1 MSA Reconstruction

The following sections discuss the configurations from different tools used in this study for MSA reconstruction.

5.1.1 MAFFT

The MAFFT is a multiple sequence alignment program based on Fast Fourier Transform [10]. There are multitude of options provided by MAFFT v7.453 to infer multiple alignments of the sequences. This study used the $--auto$ option that selects the alignment method based on the input data (simulated sequences and real sequences) that is provided. And, the $--leavegappyregions$ option to infer the MSA without gap region realignment [10]. It is important to note that, the trees τ supplied at the input is a special mafft formatted tree, which is converted from a newick tree file format. We applied these options to all data types used in this study.

5.1.2 PRANK

The phylogeny-aware method PRANK v.170427 [14] is the second tool used in this study to infer MSA. Similar to MAFFT the PRANK also offers additional options other than the tree and data (simulated and real sequence). This study used additional options $-F$ and $-once$ in PRANK. As per the PRANK manual, $-F$ is an option applied to avoid the force insertions and $-once$ is used to limit the iterations to one. This study followed the same settings mentioned above for all data types used in this study.

5.1.3 ProPIP

The third and last tool used in this study to generate MSA is called ProPIP (Chapter 3). The simulated sequences and real sequences were given as input to ProPIP along with all other parameters (trees τ, rate matrix Q, insertion rate λ and deletion rate μ), so that the progressive aligner

only has to estimate the MSA itself. Based on the ProPIP manual available at [12], the following parameters were fixed for all data types used in this study:

INDELible data

- Tree: Balanced, Symmetric tree with exponential branch lengths with tree rate 1/8 [Ref: Section 4.1].
- Number of taxa: 8
- Rate matrix: WAG01
- Insertion rate (λ) and Deletion rate (μ): Estimated from True MSA and Tree using a program implemented by Maiolo et al.[12] based on Levenberg–Marquardt algorithm.

PIP data

- Tree: Balanced, Symmetric tree with exponential branch lengths with tree rate 1/8 [Ref: Section 4.1].
- Number of taxa: 8
- Rate matrix: WAG01
- Insertion rate (λ): 10 ("true" (simulation) value)
- Deletion rate (μ): 0.05 ("true" (simulation) value)

Real data

- Tree: Generated using PRANK v.170427 using its $-treeonly$ option.
- Number of taxa: 13
- Rate matrix: WAG01
- Insertion rate (λ) and Deletion rate (μ): Estimated from True MSA and Tree using a program implemented by Maiolo et al.[12] based on Levenberg–Marquardt algorithm.

The ProPIP has different additional features as explained in Chapter 3 [12]. In this study the features from ProPIP is assessed with all data types mentioned in Chapter 4 using the corresponding parameters explained in the Section 5.1.3. The following Sections 5.1.4 to 5.1.8 explains the experiment setups used on each additional methods in ProPIP.

5.1.4 ProPIP under Discrete Gamma distribution

The aim of this method is to see how changes in Discrete Gamma distribution parameters reflect on the variation of indel length distribution. The gamma distribution is described with two parameters. The number of categories n and shape parameter, α. In detail, the number of categories split the sequences and assign different probabilities for different categories. Which helps to manipulate the sequences in specific regions. Fixing the shape parameter α is challenging as the sequence representation is unknown, therefore the typical value n= 4 is applied. Finally, several runs by fixing $n = 4$ and by varying α values i.e., α= {0.1,0.5,1.0,2.0,3.0} were experimented on default ProPIP ($k = 1$). Thus, the parameter configuration under this experiment are described as:

- $k = 1$, $n = 4$, α=0.05
- $k = 1$, $n = 4$, α=0.10
- $k = 1$, $n = 4$, α=0.25
- $k = 1$, $n = 4$, α=0.50
- $k = 1$, $n = 4$, α=2.0
- $k = 1$, $n = 4$, α=3.0

5.1.5 ProPIP under k-Factor

As explained in Section 3.3, the value of $'k'$ scales the input parameters λ and μ of our aligner, ProPIP. To assess the variation of indel length distribution in the inferred MSA and understand how closer our progressive aligner is to other popular aligners, different k values i.e., $\{k = 0.05, 0.10, 0.25, 0.5, 2.0, 3.0\}$ were tested and further studied. Note that, ProPIP with parameter k equal to 1 is the default ProPIP, which uses unscaled λ and μ parameter values.

5.1.6 ProPIP under Discrete Gamma distribution and k-Factor

This method is designed to investigate the combined effect of the k-Factor and Discrete Gamma distribution in ProPIP alignments. Several alignment scenarios were conducted to see the combined effect of both methods in the inferred MSA, specifically the dynamics of insertion and deletion events. For all alignment runs the number of categories is fixed to 4 and the other parameter are varied. Therefore, the framework of this experiment is defined as follows:

- $k = 0.05$, α=0.05 and $n = 4$
- $k = 0.05$, α=0.10 and $n = 4$
- $k = 0.10$, α=0.05 and $n = 4$
- $k = 0.10$, α=0.10 and $n = 4$

5.1.7 ProPIP under Stochastic Backtracking

To infer MSA using ProPIP under Stochastic Backtracking method, the distortion parameter or temperature parameter T has to be provided together with the parameters explained in Section 5.1.3. In this study different T values are selected randomly based on the theoretical explanation provided in Section 3.4. The values of T selected for this study are $T = \{1, 3, 5, 10\}$.

5.1.8 ProPIP under Short Time Fourier Transform

As explained in Section 3.5, to simulate ProPIP under STFT several additional parameters are need to be fixed. One of the parameters used in this study is the finite impulse response (FIR) filter or window function. There are many window functions implemented in this method. We selected randomly the 'welch' window function with window size of 128. The other parameters are input data (simulated and real sequences) and tree τ, for which we used the default settings explained in Section 5.1.3.

Understanding the dynamics of indels helps to understand the changes happening in the evolutionary history of the species. This study is generally interested in the gap length information from the observed MSA and different strategies to tune them. For this, after the construction of multiple sequence alignments (MSAs) from individual sequences (simulated and protein sequences) using the above mentioned tools and frameworks, an indel based study, quality check of MSAs and an individual MSA analysis is conducted. The following sections explains the aforementioned experiments.

5.2 Indel length distribution

Information regarding the dynamics of indels (insertions and deletions) plays a vital role to understand the evolutionary divergence among the species. This study start by extracting the length of the gaps per taxa on each MSA (reference and observed MSA). For example, in a sequence $--AC--T-AAG---$ the length of gaps is saved in a list as $[2, 2, 1, 3]$. This process is applied globally to input (simulated and protein alignments) and output (inferred alignments) data by saving the indel length information in separate lists. In the next step, using these lists, the indel length distribution of the reference MSA and observed MSA is plotted using a log-log bar graph. Finally, to understand how the statistical values differ on each datasets the summary statistics is collected in a table and further analysed.

5.3 Indel block distribution

The study further extends to a different approach which excludes the single characters placed in between the gaps in a sequence on each reconstructed MSA. This study refer this method as Indel block method. For instance, the aforementioned sequence in Section 5.2 under indel block method would produce a list of values $[2, 3, 3]$ instead of $[2, 2, 1, 3]$. This explains the effect of misplaced single characters in an indel length distribution. To explain this method the corresponding log-log plots are plotted together with indel length distribution and reference indel length distribution. Note that, the id represents the indel length distribution and Xid represents the Indel block distribution in this report.

5.4 Quality check using qscore

To analyse the quality of the alignments inferred using MAFFT v7.453, PRANK v.170427 and ProPIP, four different quality scores were calculated using the standard qscore [19] software. The description of the scores are exactly the same as explained in the previous Chapter, Section 3.6.3.

5.5 Analysis using Pixel Plot

A large set of MSA visualization tools are available today. In order to compare and understand where the inferred MSA agree on each other with respect to the reference MSA, a visulisation tool is considered. SuiteMSA [1] is a visualisation tool, which provides multitude of options to do alignment comparisons. In this study to visualise and understand how indels are dispersed on each MSA, the Pixel Plot method from SuiteMSA is used. To select the MSAs for visual interpretation, after the Quality check using the standard qscore tool we selected the observed MSAs having higher deviation of Q Score values from the reference MSA and we plotted these MSA's side by side and further studied.

Chapter 6

Results and Discussions

This Chapter is divided into 3 main sections based on the data types that we used in this study. The Sections 6.1, 6.2 and 6.3 explains the results obtained for the MSA evaluation methods explained in Section 5.2 to Section 5.5 for INDELible, PIP and real data with corresponding summary statistics table, log-log plot and the Pixel-plot. The summary statistics table provides the information such as; number of indels (nIndels), maximum indel length (Max-IL), mean, median and standard deviation (SD) of the appropriate dataset. The log-log plot shows the 'true' indel length distribution, inferred indel length distribution and inferred indel block distribution (distribution obtained from indel block method See Section 5.3). The Pixel Plot provides a visual comparison of a 'true' MSA with corresponding inferred MSA's. All computations in ProPIP is carried out in High Performance Cluster at ZHAW. The Matlab scripts used to produce these results and additional pixel plots generated using SuiteMSA are provided at the Appendix part of this report.

6.1 INDELible data analysis

6.1.1 Dynamics of indels in MAFFT, PRANK and ProPIP

This section explains the comparative results obtained from MAFFT, PRANK and ProPIP using a summary statistics table (Table 6.1), a log-log plot (Figure 6.2), and a Pixel plot (Figure 6.1).

Firstly, the table 6.1 shows the summary statistics of the 'true' and inferred indel length distribution. Overall, the traditional aligners PRANK v.170427 and MAFFT v7.453 outperformed our aligner ProPIP. To begin with, we compare the three aligners using the number of indels and maximum indel length present in each of these aligners. The number of indels present in the 'true' alignments of multiple sequences from INDELible was 11511 with a maximum indel length of 37 and an average indel length of 3.116. On the one hand, the MAFFT underestimated the number of events occurring and over-aligns the sequences, which results in shorter MSA's and lower indel events (9279). However, it is interesting to note the presence of long indels in MAFFT (a maximum indel length of 45). On the other hand, the PRANK v.170427 inferred a closer number of indels (10226) to the 'true' number of indels (11511), which also includes longer indels (a maximum indel length of 38) similar to MAFFT. In contrast, our aligner is based on a single indel model therefore we expect more point mutations thus the alignments from our tool resulted with larger number of indels (19539) which agree with the underlying model PIP that our tool is using. But interestingly we noted that our model is able to mimic the long indels in some alignments with a maximum indel length of 30.

Secondly, the log-log plot (Figure 6.2) shows the frequency of the indel lengths distributed in the 'true' and inferred MSA's. The log-log plot confirms the above interpretations. To elaborate, in Figure 6.2 the blue bar represents the indel length distribution of the 'true' data, the red bar represents the inferred indel length distribution and the yellow bar denotes the inferred indel length distribution which resulted from the method explained in the Section 5.3. From the log-log plot, we have manually observed that our tool ProPIP assigns more single indel events compared to other state of the art aligners MAFFT and PRANK. Also, comparing MAFFT and PRANK using the log-log plot we could say that PRANK is inferring more longer indels (based on the region where 'true' indel length is around 30) than MAFFT. At the same time ProPIP failed to infer the indels in this same region. We manually visualized certain regions in the alignments where ProPIP splits

the longer indels into several shorter indels using single character residues. This leads us to the indel block method. The distribution obtained from Indel block method confirms the misplacement of single characters in between the long indels. We further tried to visually understand the indel patterns using an MSA visualization tool called Pixel-Plot. In Figure 6.1, for the particular MSA that we used in this plot, the MAFFT and PRANK were able to achieve same MSA length with almost similar conserved regions present in the 'true' MSA of length 224 (with a small difference), whereas our aligner inferred a lengthy alignment (of length 318) with extremely large number of single indel events.

To conclude, we have observed that, in contrast to other traditional aligners, our aligner ProPIP shows an extremely different indel placements with long indel data. However, we further analyzed the additional features implemented in our tool to see if it is possible to tune the output of our tool with the same data.

(100,8)	True(id)	MAFFT v7.453		PRANK v.170427		ProPIP	
		id	Xid	id	Xid	id	Xid
nIndels	11511	9279	8971	10226	9692	19539	14418
Max-IL	37	45	45	38	38	30	33
Mean	3.116	3.654	3.780	3.306	3.488	2.041	2.767
Median	1	2	2	2	2	1	1
SD	3.850	4.394	4.506	3.307	3.489	2.132	3.321

Table 6.1: The summary statistics of the 'true' Indel length distribution of INDELible data (True(id)) is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by MAFFT v7.453, PRANK v.17042, and ProPIP. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution

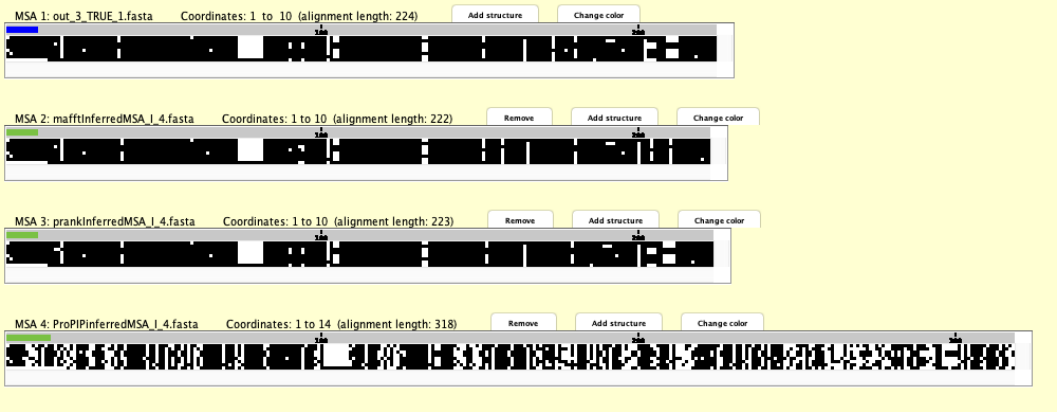


Figure 6.1: The Pixel Plot[1] (Section 5.5). A simulated 'true' MSA using INDELible is compared with MSA's generated by MAFFT v7.453 (MSA 2), PRANK v.17042 (MSA 3), and ProPIP (MSA 4). Note: black pixel represents Characters and white pixel represents Indels.

6.1.2 Dynamics of indels in ProPIP under Discrete Gamma distribution

This section illustrates the results obtained with the additional feature Discrete Gamma distribution from ProPIP (as explained in the Section 3.2) using a summary statistics table (Table 6.2), a log-log plot (Figure 6.3), and a Pixel plot (Figure 6.4).

The experiment using Discrete Gamma distribution showed a slight improvement of the indel events present in the MSA reconstructed using ProPIP. In detail, we tested the effect of keeping n= 4 and varying the α parameter from the Discrete Gamma distribution for the same parameter (λ and μ) values of the ProPIP used in the Section 6.1.1, which we further call as the 'default ProPIP' in this section and the following sections. The number of indels present in the alignments inferred using default ProPIP (Table 6.1) was 19539 with a maximum indel length of 30 and an average indel length of 2.041. We observed a gradual decrease in the number of indels as we decrease the value of α. On the one hand, the value of α= 3 generated a similar statistic as that of the default

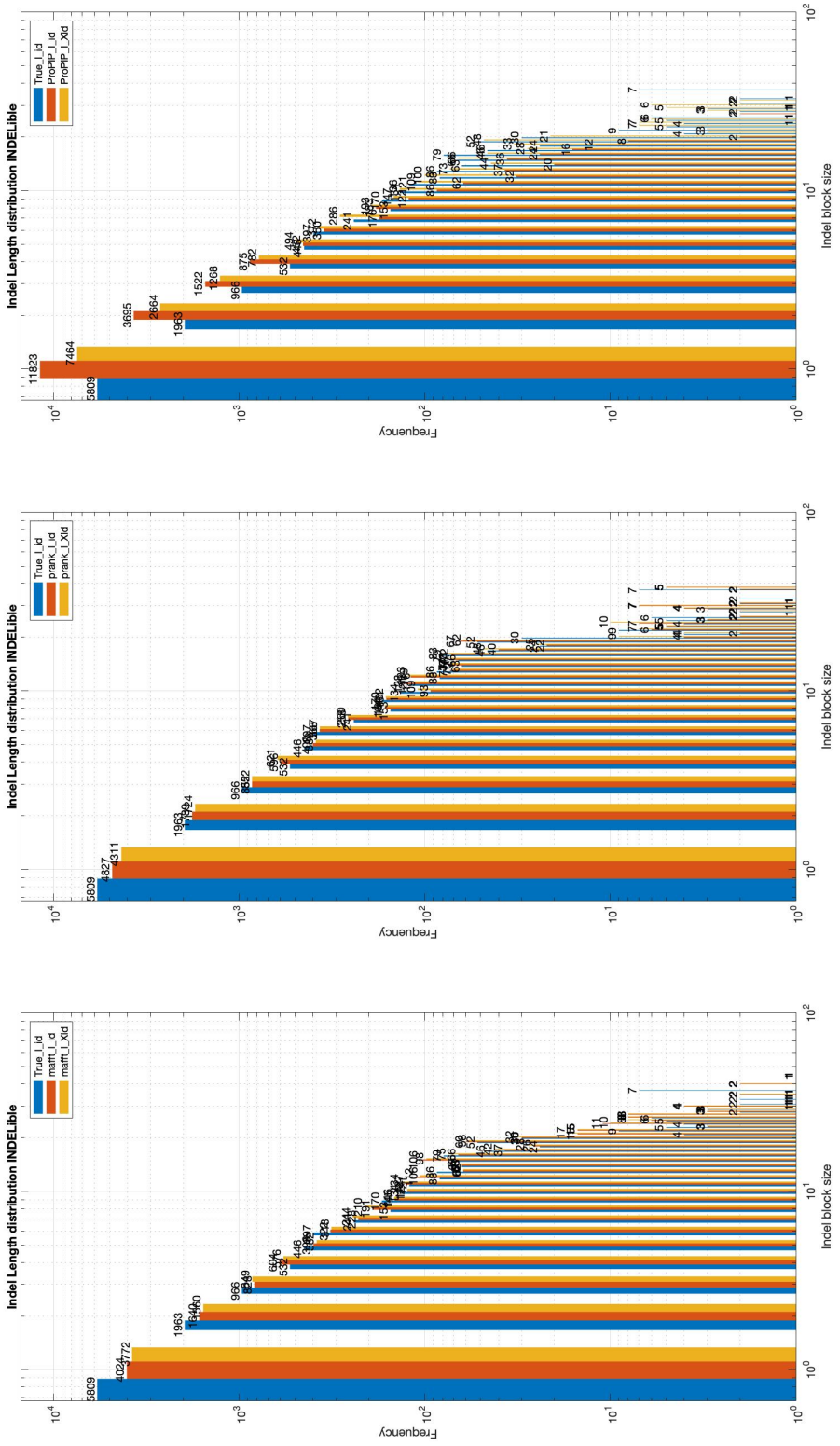


Figure 6.2: The Log-Log Plot. The 'true' Indel length distribution of INDELible data is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by MAFFT v7.453, PRANK v.17042, and ProPIP. Note: The legend blue represents 'true' Indel length distribution, red represents inferred indel length distribution and yellow represents the inferred indel block distribution.

ProPIP. And, on the other side, where α= 0.10 the number of indels went down to slightly greater than 18000. However, these variation of indel events seems to be not useful for further analysis.

In order to confirm this result, we used the same structure of log-log plot as in the previous section (Figure 6.2) and used the indel length data generated using the framework of ProPIP under Discrete Gamma distribution (See Section 5.1.4). In the log-log plot, the dynamics of indel length distribution is apparently visible (Figure 6.4) and agrees with the above mentioned conclusion. Also, the regions where indel length is around 30 remains undetected using ProPIP under Discrete Gamma distribution. This makes it clear that tuning of long indel data is possible only in some extent using the α values that we used. We further visualized the same MSA, used in Figure 6.1, under this experiment results. In Figure 6.4, the MSA 4 represents the alignment inferred using the default ProPIP and MSA 5 to MSA 8 indicates the MSA's resulted from this experiment. The MSA reconstructed using default ProPIP in this plot is having higher MSA length (of length 318) compared to other inferred MSA's. However, under Discrete Gamma distribution the MSA length is decreased when α is relatively smaller but, the indel placements doesn't agree with the 'true' MSA in most of the cases.

In sum, ProPIP under Discrete Gamma distribution showed the presence of short indels in the reconstructed MSA's, which is same as that of in default ProPIP. Since the value of the parameter α is selected as random, a finite conclusive result cannot be made from these experiments. However, based on the variation in the MSA length and decrease in short indels as witnessed from log-log plot and Pixel plot we could say that to some extent it is possible to tune the default ProPIP output alignment using ProPIP under Discrete Gamma distribution feature with long indel simulated MSA's.

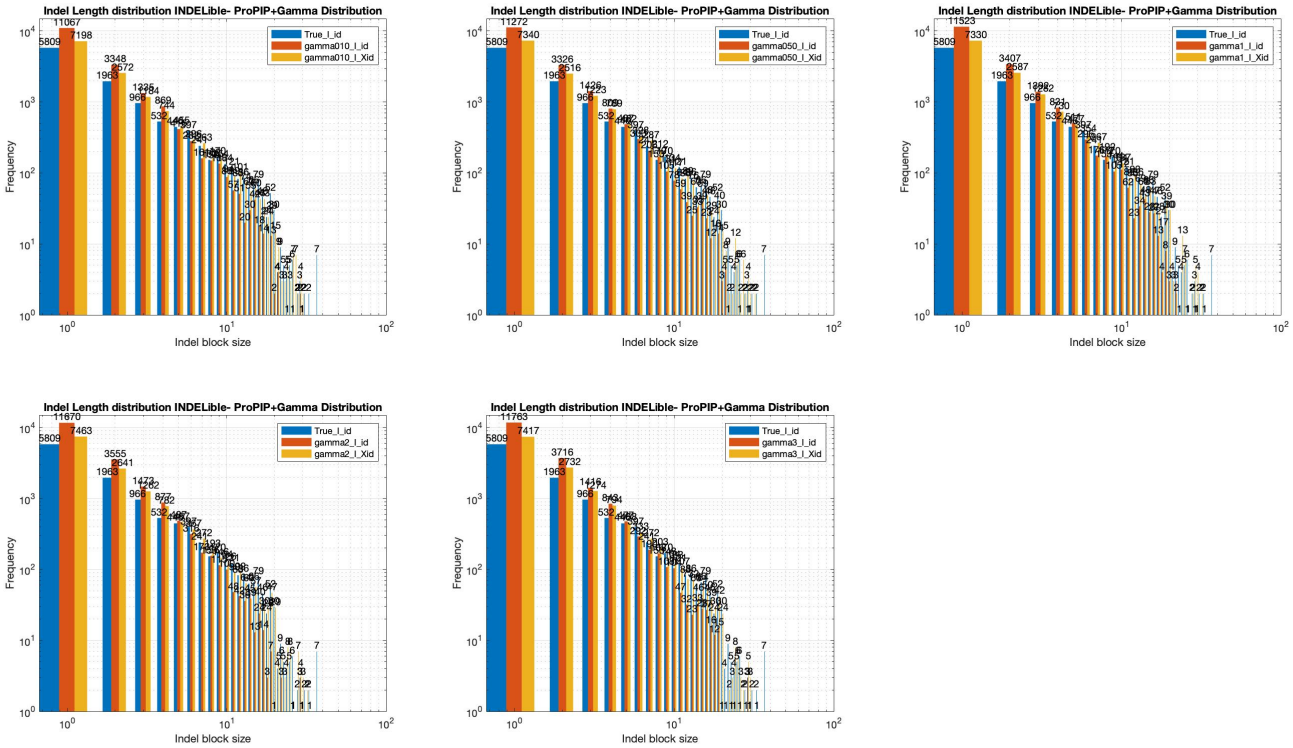


Figure 6.3: The Log-Log Plot. The 'true' indel length distribution of INDELible data is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by ProPIP with k=1 under Gamma n=4, α=0.10, ProPIP with k=1 under Gamma n=4, α=0.50 , ProPIP with k=1 under Gamma n=4, α=1, ProPIP with k=1 under Gamma n=4, α=2, and ProPIP with k=1 under Gamma n=4, α=3. Note: The legend blue represents 'true' Indel length distribution, red represents inferred indel length distribution and yellow represents the inferred indel block distribution.

(100,8)	True (id)	G0.10		G0.50		G1		G2		G3	
		id	Xid	id	Xid	id	Xid	id	Xid	id	Xid
nIndels	11511	18108	13700	18435	13964	18776	14060	19163	14283	19318	14346
Max-IL	37	30	30	30	32	30	33	30	33	30	33
Mean	3.116	2.045	2.703	2.050	2.707	2.043	2.728	2.023	2.723	2.026	2.728
Median	1	1	1	1	1	1	1	1	1	1	1
SD	3.850	2.1692	3.256	2.1604	3.228	2.148	3.269	2.105	3.290	2.142	3.255

Table 6.2: The summary statistics of the 'true' Indel length distribution of INDELible data (True(id)) is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by ProPIP with k=1 under Gamma n=4, α=0.10, ProPIP with k=1 under Gamma n=4, α=0.50 , ProPIP with k=1 under Gamma n=4, α=1, ProPIP with k=1 under Gamma n=4, α=2, ProPIP with k=1 under Gamma n=4, α=3. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

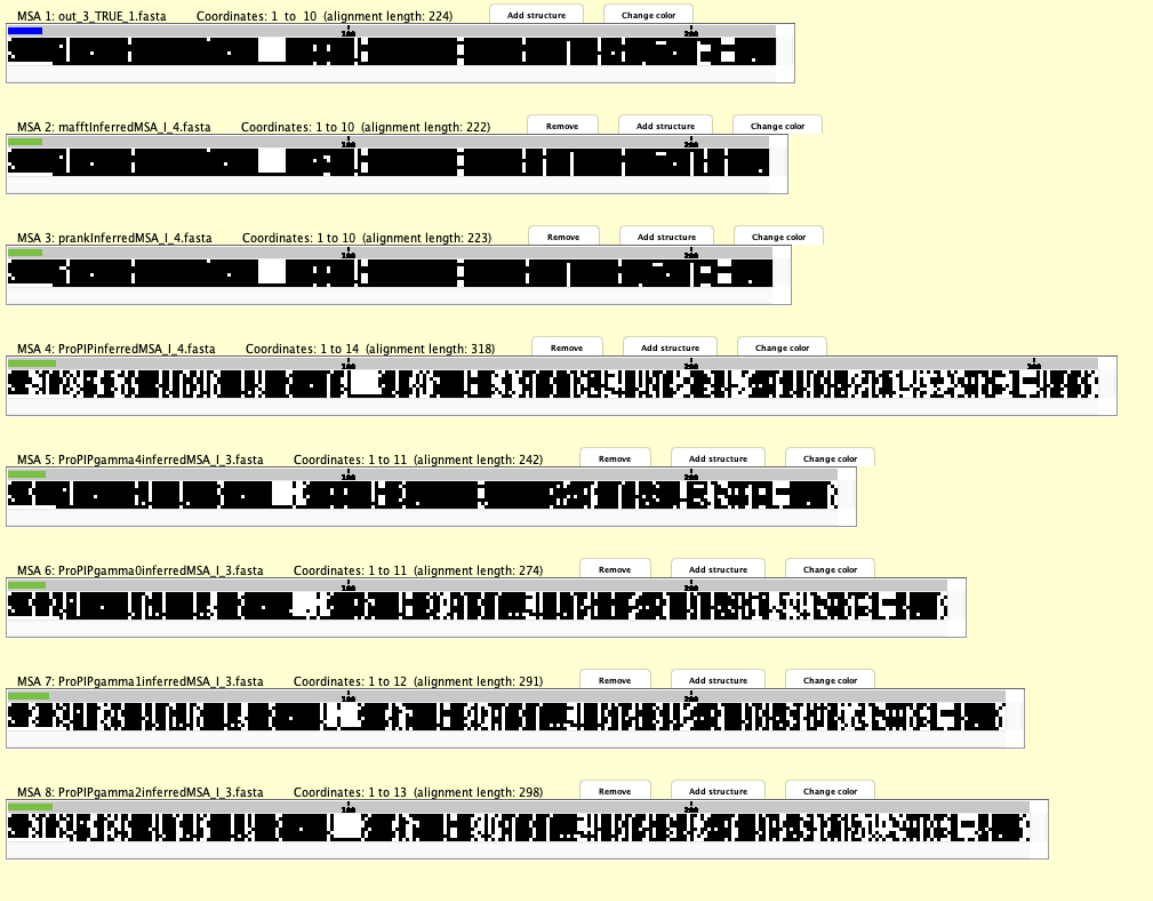


Figure 6.4: The Pixel Plot[1] (Section 5.5). A simulated 'true' MSA using INDELible is compared with MSA's generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), ProPIP with k=1 under Gamma n=4, α=0.10 (MSA 5), ProPIP with k=1 under Gamma n=4, α=0.50 (MSA 6), ProPIP with k=1 under Gamma n=4, α=1 (MSA 7), and ProPIP with k=1 under Gamma n=4, α=2 (MSA 8). Note: black pixel represents Characters and white pixel represents Indels.

6.1.3 Dynamics of indels in ProPIP under k-Factor

The new method using k-Factor generated more consistent and satisfying results from ProPIP. Results obtained using this new feature is elaborated through this section using a summary statistics table (Table 6.3), a log-log plot (Figure 6.6), and a Pixel plot (Figure 6.5).

The research using k-Factor showed an improved dynamics of the indel events in the inferred MSA's compared to the default ProPIP results given in Section 6.1.1. In detail, we tested the dynamics of indels under different k values in default ProPIP. The number of indels present in the inferred MSA's using default ProPIP (Table 6.1) was 19539 with a maximum indel length of 30 and an average indel length 2.041. On the one hand, by doubling the parameter values of default

ProPIP (same as k= 2) we received a high hike in the number of indels (25071) with an average indel length of 2.090 and maximum indel length of 40. These values further increased as we moved the value of k to 3. On the other hand, more interesting and encouraging results showed up as we decreased the value of k to less than 1. For instance, for a very low value at k= 0.05, the ProPIP inferred alignments generated a closer indel length distribution compared to the 'true' indel length distribution with 13725 indels and maximum indel length of 30. As mentioned earlier our aligner ProPIP is based on a single indel model, but we witnessed the possibility of tuning our model to infer long indels.

The log-log plot of this experiment confirms the overall improvement of the MSA's reconstructed using ProPIP under k-Factor, especially the long indels at the region where indel length is around 30. With the same structure of log-log plot (Figure 6.2), we plotted the indel length information that is extracted from the MSA's generated under this experiment. In Figure 6.6, the indel length distribution using the methods explained in Section 5.2 and 5.3 is plotted against the 'true' indel length distribution (represented with blue bars). From the log-log plot we observed that, for a higher value of k our aligner ProPIP infers more number of single indel events (when k is 3 the number of single indel event is 17598). Furthermore, it is also visible from the plot that extremely longer indels (above 40) are inferred when the k value is tuned above 1. However, if we look at the region where indel length is around 30 then the lower k value seems to be balancing the number of short and long indels. Thus, reducing the distance to 'true' Indel length distribution.

We visualized the same reference alignment as we used in Figure 6.1 for comparisons under this experiment using Pixel plot. In the Pixel plot (Figure 6.5), the MSA 4 to MSA 9 shows the MSA's reconstructed using ProPIP under k-Factor, where MSA 8 is the alignment inferred using k=1 i.e., default ProPIP. For this sample alignment, compared to other state of art aligners MAFFT and PRANK, we were able generate MSA's of similar lengths to 'true' alignment, differ in small margin. As can be seen from the Figure 6.5 that, our tool achieved more cleaner regions with characters and similar columns with gaps, especially the head portion of the MSA. This pattern of indel placements are new to our tool and providing more information regarding the optimal alignments and parameter settings is an undergoing work and out of this report.

In sum, our tool achieved a closer number of indels to 'true' under this method when the value of the parameter k is relatively smaller. Since the optimal k selection is not a topic of this research and an ongoing work, a more detailed result cannot be provided in this report. Overall, based on the improvement achieved in the number of indels, MSA length and indel placements (as witnessed from log-log plot and Pixel plot) we could say that it is possible to tune a single indel model to infer long indels.

	True	**k0.05**		**k0.10**		**k0.25**		**k0.50**		**k2**		**k3**	
(100,8)	**(id)**	id	Xid	id	Xid	id	Xid	id	Xid	id	Xid	id	Xid
nIndels	11511	13725	10466	13745	10618	14562	11060	16030	12231	25071	17307	30713	20091
Max-IL	37	30	30	30	30	30	30	30	32	43	105	37	118
Mean	3.116	2.119	2.779	2.161	2.797	2.140	2.817	2.102	2.755	2.090	3.028	2.1663	3.312
Median	1	1	1	1	1	1	1	1	1	1	2	1	2
SD	3.850	2.207	3.316	2.261	3.352	2.226	3.376	2.255	3.294	2.230	3.985	2.347	4.740

Table 6.3: The summary statistics of the 'true' Indel length distribution of INDELible data (True(id)) is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by ProPIP with k=0.05, ProPIP with k=0.10, ProPIP with k=0.25, ProPIP with k=0.50, ProPIP with k=2, ProPIP with k=3. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

	k=0.05 (frequency)		**k=0.10 (frequency)**	
Indel-length	α=0.05	α=0.10	α=0.05	α=0.10
1	8280	8191	8481	8276
2	2483	2458	2569	2533
3	1030	1042	1124	1154
4	688	717	713	783

Table 6.4: The indel length variation (only the initial section of log-log plot (Figure 6.8)) when tuned using k and α.

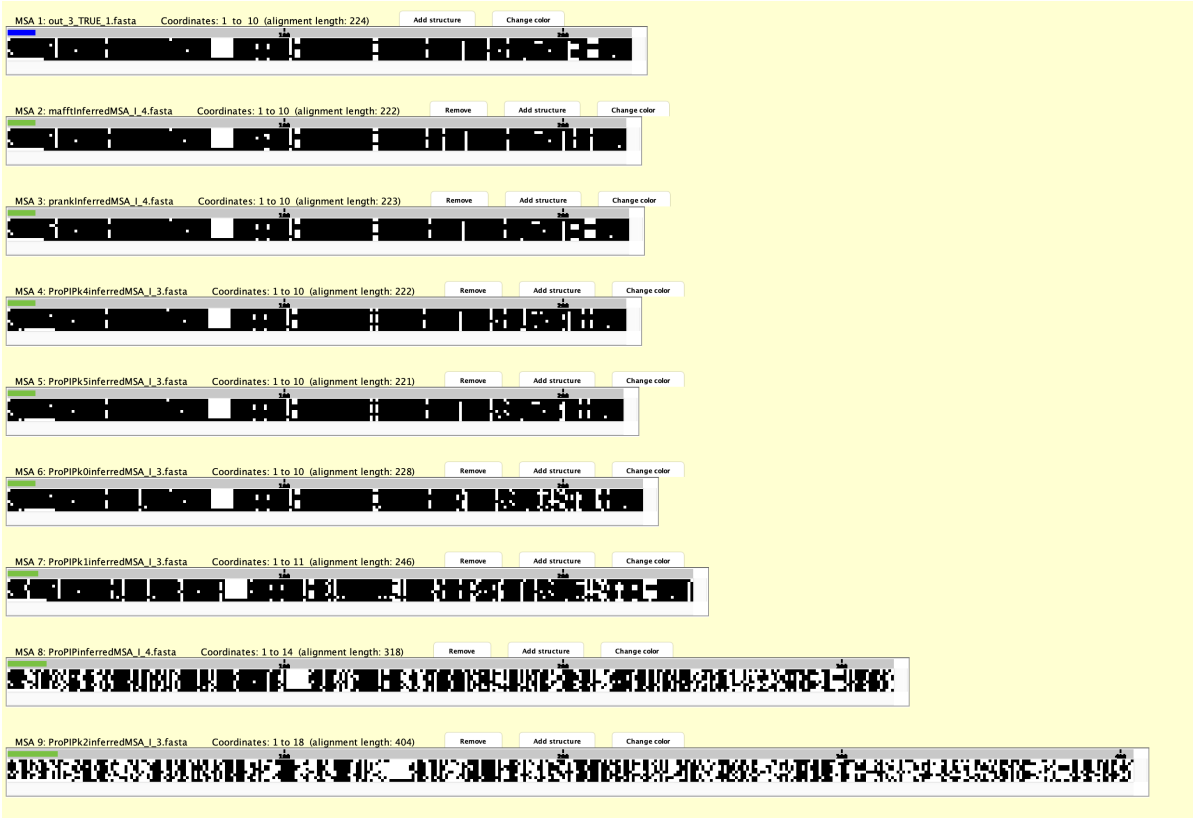


Figure 6.5: The Pixel Plot[1] (Section 5.5). A simulated 'true' MSA using INDELible is compared with MSA's generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=0.05 (MSA 4), ProPIP with k=0.10 (MSA 5), ProPIP with k=0.25 (MSA 6), ProPIP with k=0.50 (MSA 7), ProPIP with k=1 (MSA 8), and ProPIP with k=2 (MSA 9). Note: black pixel represents Characters and white pixel represents Indels.

6.1.4 Dynamics of indels in ProPIP under Discrete Gamma distribution and k-Factor

The result of k-Factor and Discrete Gamma distribution applied simultaneously in ProPIP shows that the k-Factor method is independent enough to outperform the default ProPIP results. This section elaborate the results of this method with the help of a summary statistics table [Table 6.5), a log-log plot (Figure 6.8), and a Pixel plot (Figure 6.7).

The test using both k-Factor and Discrete Gamma distribution under ProPIP showed an improved version of Discrete Gamma distribution under ProPIP results (as explained in Section 6.1.2). To elaborate, we considered the small values from k and α and created a framework to run these parameters using default ProPIP settings to understand the MSA's inferred using both methods together. As said earlier, compared to the statistics obtained using default ProPIP (Table 6.1)) and results from Discrete Gamma distribution method (Table 6.2) we witnessed relatively lower number of indel events (lowest 13796 and highest 14075). However, we believe this effect is due to the lower k that we used in this study. To explain this, for instance, we have considered the initial portion of the indel length distribution i.e., indel length from 1 to 4 (See Table 6.4). In this region we spotted a pattern which tunes the frequency of indel lengths by k and α. To be precise, when k is kept constant (either 0.05 or 0.10 in this experiment) and α is varied simultaneously it is seen that the frequency of indel length 1 and 2 is decreasing and 3 and 4 is increasing. However, on the other hand, when α is kept constant (either 0.05 or 0.10) and value of k-Factor is varied simultaneously, we only observed either a decrease or increase of frequency at the same time depending on the k value variation (increase or decrease). That is, when k is decreased the frequency of indel lengths 1,2,3 and 4 is also decreasing and vice versa. We made conclusion from this pattern that k-Factor is independent enough to make changes in the default ProPIP alignments, for this particular datatype.

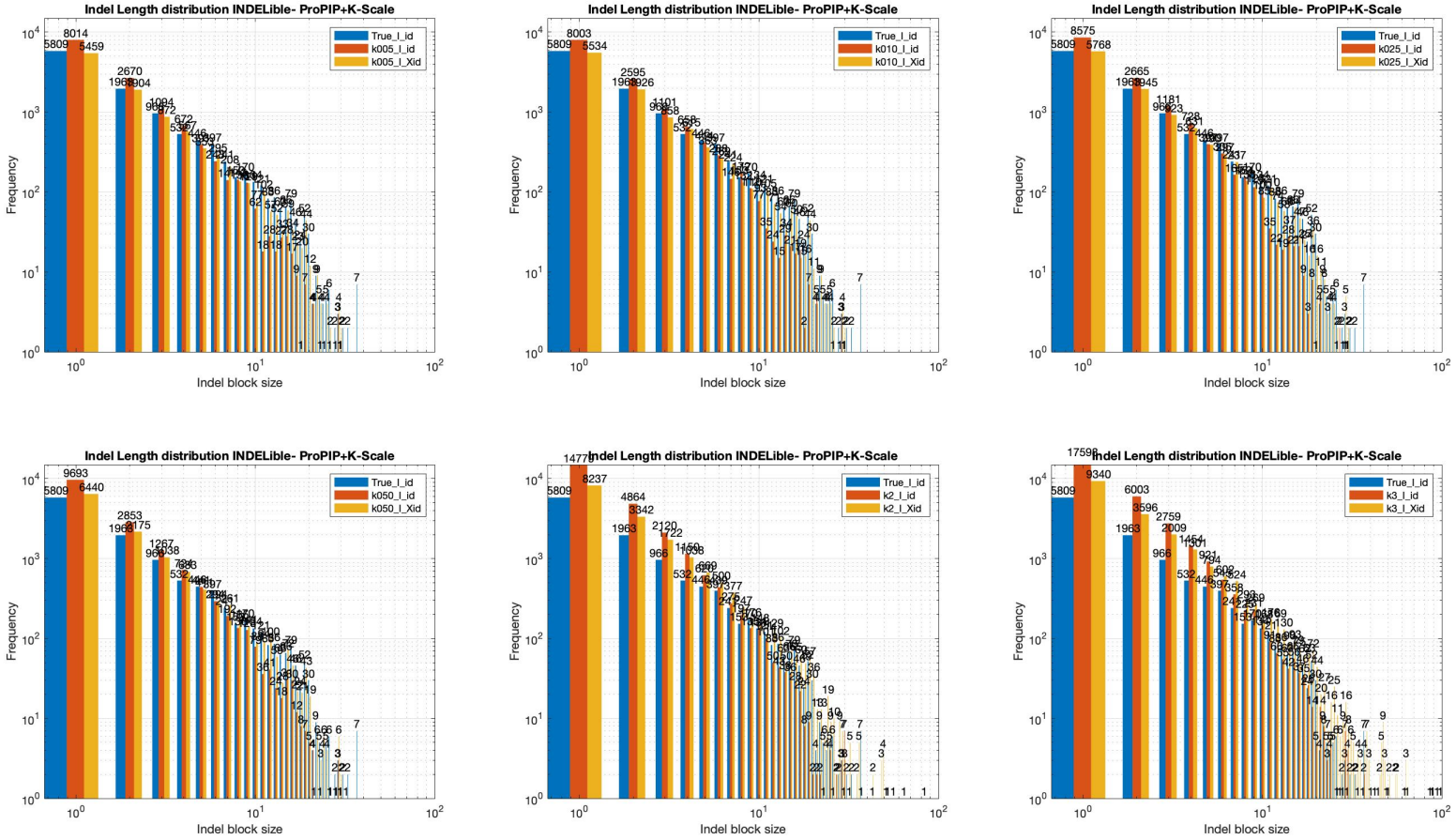


Figure 6.6: The Log-Log Plot. The 'true' Indel length distribution of INDELible data is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by (from top-left to bottom-right) ProPIP with k=0.05, ProPIP with k=0.10, ProPIP with k=0.25, ProPIP with k=0.50, ProPIP with k=2, and ProPIP with k=3. Note: The legend blue represents 'true' Indel length distribution, red represents inferred indel length distribution and yellow represents the inferred indel block distribution.

We further studied the MSA's using the visualization tool, SuiteMSA. The Pixel plot of the same reference MSA as we used in Figure 6.1 is again considered in this experiment. From the Pixel plot (Figure 6.7), the MSA 4 represents the alignment reconstructed under default ProPIP and MSA 5 to MSA 8 are the alignments inferred using this method. In this method compared to other aligners MAFFT and PRANK, for instance, we were able to generate MSA's of similar lengths to 'true' MSA when k=0.10 and α=0.05 (MSA 7). Also, in MSA 7 similar to true alignment we found same pattern of indel placements, differ in some regions at the tail of the MSA. To summarize, more motivating and supporting results for the new k-Factor feature is achieved through the ProPIP under Discrete Gamma distribution and k-Factor method.

	True	kg0505		kg0510		kg1005		kg1010	
(100,8)	(id)	id	Xid	id	Xid	id	Xid	id	Xid
nIndels	11511	13796	10569	13724	10469	14238	10910	14075	10794
Max-IL	37	30	31	30	30	30	30	30	30
Mean	3.116	2.121	2.768	2.130	2.791	2.120	2.767	2.145	2.7970
Median	1	1	1	1	1	1	1	1	1
SD	3.850	2.275	3.307	2.258	3.311	2.232	3.287	2.276	3.321

Table 6.5: The summary statistics of the 'true' Indel length distribution of INDELible data (True(id)) is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by ProPIP with k=0.05 under Gamma n=4, α=0.05, ProPIP with k=0.05 under Gamma n=4, α=0.10, ProPIP with k=0.10 under Gamma n=4, α=0.05, ProPIP with k=0.10 under Gamma n=4, α=0.10. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

Figure 6.7: The Pixel Plot[1] (Section 5.5). A simulated 'true' MSA using INDELible is compared with MSA's generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), ProPIP with k=0.05 under Gamma n=4, α=0.05 (MSA 5), ProPIP with k=0.05 under Gamma n=4, α=0.10 (MSA 6), ProPIP with k=0.10 under Gamma n=4, α=0.05 (MSA 7), and ProPIP with k=0.10 under Gamma n=4, α=0.10 (MSA 8), Note: black pixel represents Characters and white pixel represents Indels.

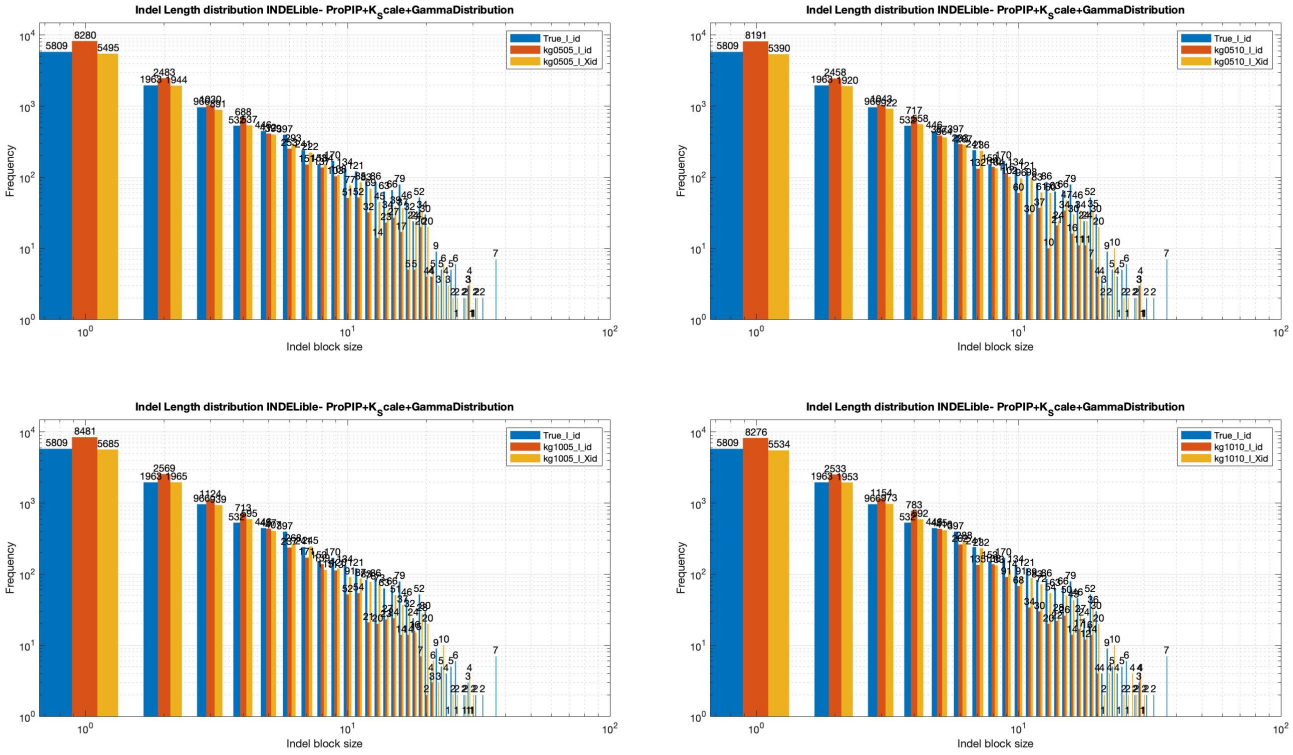


Figure 6.8: The Log-Log Plot. The 'true' Indel length distribution of INDELible data is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by (from top-left to bottom-right) ProPIP with k=0.05 under Gamma n=4, α=0.05, ProPIP with k=0.05 under Gamma n=4, α=0.10, ProPIP with k=0.10 under Gamma n=4, α=0.05, and ProPIP with k=0.10 under Gamma n=4, α=0.10. Note: The legend blue represents 'true' Indel length distribution, red represents inferred indel length distribution and yellow represents the inferred indel block distribution.

6.1.5 Dynamics of indels in ProPIP under Stochastic Backtracking

The alignments reconstructed using ProPIP under SBDP method as explained in Section 3.4 produced overestimated indel lengths with large chunk of single indel events. This section elaborate the results of this method through a summary statistics table [Table 6.6), a log-log plot (Figure 6.10), and a Pixel plot (Figure 6.9).

The SBDP algorithm in ProPIP produced insufficient results for this study among all the features tested in ProPIP. As explained in the Section 3.4, we have run the distortion parameter T at 4 different values as explained in Section 5.1.7. Applying T parameter in the default ProPIP settings helps to randomize the sub-optimal MSA's selected using SB algorithm. As per the algorithm, the smaller T value selects the maximum distortion probability state by penalizing the other two possible states (the three possible states: *match*, *gapX* and *gapY* ref:Chapter 3 in [16]) and behaves as a greedy algorithm. Whereas, the larger T value generates more randomized sub-optimal MSA's allowing equal chance for the sub-alignments i.e., less greedy.

Overall, the results obtained from this experiment agree with the SB algorithm. In detail, the indel length distributions received from the output alignments of this experiment is plotted as bar graphs in log-log scale. From the log-log plot, we observed a boost in the frequency of all indel lengths in every scenarios tested under SBDP. Also, we obtained large chunk of indel lengths in the MSA's with a maximum indel length of 380 when T=10. Furthermore, when T=5 the number of indels (64539) become very small (smallest among the experiments conducted under this method) with maximum indel length of 144. It is important to note that, even though the generation of alignments is based on its likelihood value (even under SBDP), the likelihood to get select these sub-optimal alignments are controlled using the distortion parameter T. Therefore, the larger T

makes all the likelihood values an equal chance to get selected.

The Pixel plot showed the rapid change of indels in the alignments when compared using the same 'true' MSA as we used in Figure 6.1. In Figure 6.9, we compared the inferred alignments using a reference MSA (MSA1). The MSA 4 represents the alignment obtained using default ProPIP and MSA 5 to MSA 8 represents the output alignments generated using this method. In this particular case all the scenarios other than T=1 produced relatively longer MSA's with widely scattered indels compared to default ProPIP alignment.

In conclusion, we gained useful results which agrees with the SB algorithm. However, the results obtained are not helping to make inferences for this study.

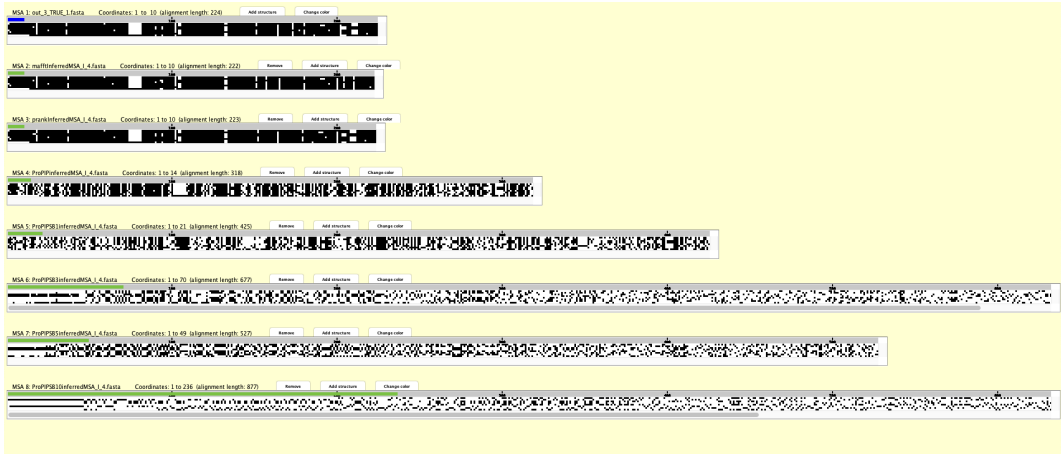


Figure 6.9: The Pixel Plot[1] (Section 5.5). A simulated 'true' MSA using INDELible is compared with MSA's generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), ProPIP with k=1 under SBDP with T= 1 (MSA 5), ProPIP with k=1 under SBDP with T= 3 (MSA 6), ProPIP with k=1 under SBDP with T= 5 (MSA 7), and ProPIP with k=1 under SBDP with T= 10 (MSA 8). Note: black pixel represents Characters and white pixel represents Indels.

(100,8)	True (id)	T1		T3		T5		T10	
		id	Xid	id	Xid	id	Xid	id	Xid
nIndels	11511	87799	37004	71834	29594	64539	25934	72090	26998
Max-IL	37	48	91	47	133	144	215	380	383
Mean	3.116	2.410	5.717	2.730	6.627	2.979	7.414	3.374	9.010
Median	1	2	4	2	3	2	4	2	4
SD	3.850	2.150	6.367	2.579	8.797	3.226	10.567	7.343	16.754

Table 6.6: The summary statistics of the 'true' Indel length distribution of INDELible data (True(id)) is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by ProPIP with k=1 under SBDP with T= 1, ProPIP with k=1 under SBDP with T= 3, ProPIP with k=1 under SBDP with T= 5, ProPIP with k=1 under SBDP with T= 10. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

(100,8)	True (id)	w128	
		id	Xid
nIndels	11511	24670	17336
Max-IL	37	81	164
Mean	3.116	2.280	3.245
Median	1	1	2
SD	3.850	3.404	6.297

Table 6.7: The summary statistics of the 'true' Indel length distribution of INDELible data (True(id)) is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by ProPIP with k=1 under STFT with fiter: welch and filter size: 128. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

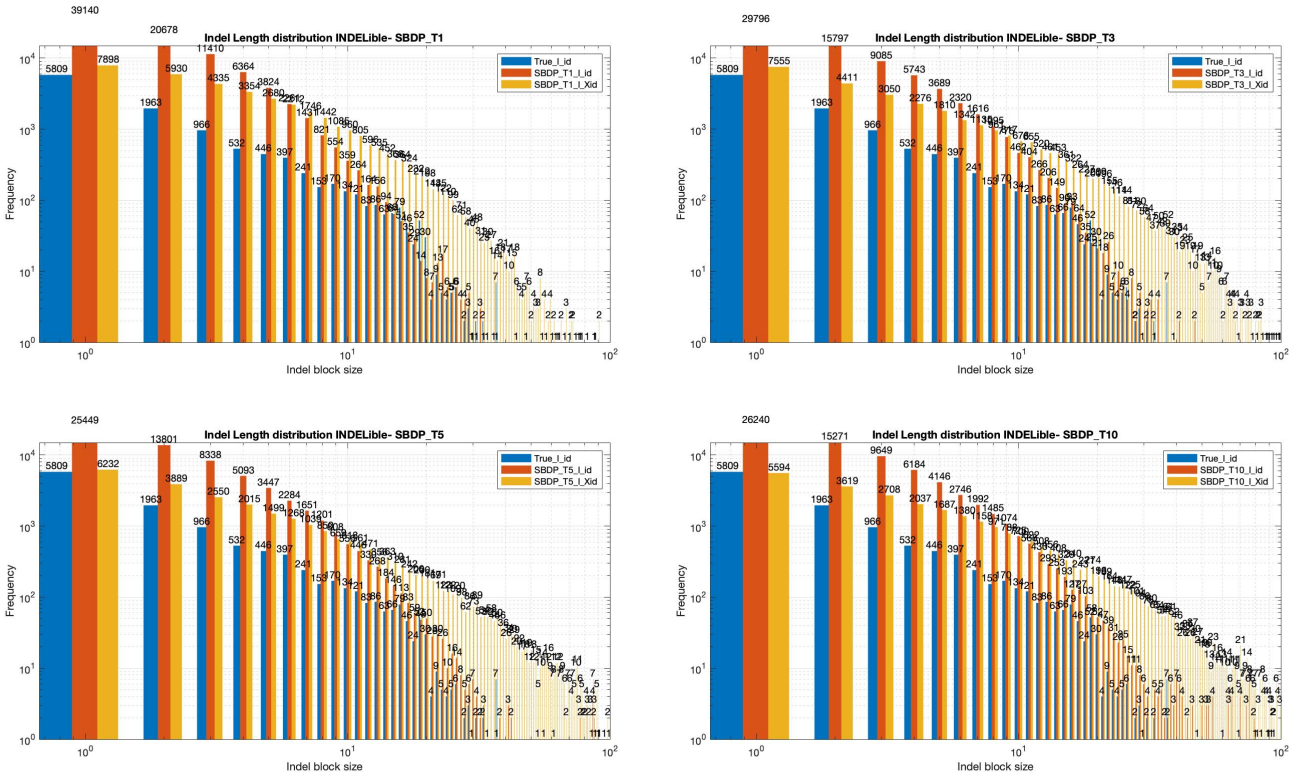


Figure 6.10: The Log-Log Plot. The 'true' Indel length distribution of INDELible data is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by (from top-left to bottom-right) ProPIP with k=1 under SBDP with T= 1, ProPIP with k=1 under SBDP with T= 3, ProPIP with k=1 under SBDP with T= 5, and ProPIP with k=1 under SBDP with T= 10. Note: The legend blue represents 'true' Indel length distribution, red represents inferred indel length distribution and yellow represents the inferred indel block distribution.

6.1.6 Dynamics of indels in ProPIP under Short Time Fourier Transform

This section explains the experiment results achieved using Short-Time Fourier Transform method from ProPIP based on *welch* FIR filter with filter size 128. It is apparent that the STFT method is not a good alignment tuning feature for ProPIP when used under INDELible data. The summary statistics table (Table 6.7), a log-log plot (Figure 6.12), and a Pixel plot (Figure 6.11) is used to elaborate this.

First of all, the STFT analysis produced less precise results with lot of indels and long chunk of indel lengths. To start with, we tested the STFT method using welch window with window size 128 (implemented by Maiolo et al., see Chapter 5 in [16]) to compare the alignments produced with other features implemented in ProPIP. After generating alignments using STFT, we collected the indel length distribution values using methods explained in Section 5.2 and 5.3. Then, we calculated the summary statistics and plotted the log-log plot as in every other test conducted under ProPIP in this work. As mentioned earlier the default ProPIP inferred 19539 indels with a maximum indel length of 30 and an average indel length 2.041. Compared to this, the STFT method produced more than 24000 indels with a maximum indel length of 81 (see Table 6.7). We visually compared the log-log plot which is plotted against the 'true' indel length distribution. To illustrate, when comparing the count of each indel length in the 'true' Indel length distribution the STFT method inferred a greater number of indels in most of the instances. Also, when we look at the tail of the log-log distribution plot we can see that the STFT produced a higher number of lengthier indels in the region where indel lengths are around 30 and above. To interpret the indels inferred in the alignments we visually compared them with a reference MSA (same as in Figure

6.1) using the Pixel plot. The Pixel plot of this particular MSA makes it clear that the STFT is inferring shorter indels in most of the regions in the alignment which leads to lengthy MSA's.

To summarize, after conducting a test under STFT method we have seen distinct results from it compared to true and default ProPIP. Due to time constrains we tested only with *welch* window, however, there are other window functions already implemented in this method. Therefore, the STFT method require more trials to make inferences about tuning default ProPIP to infer long indels.

Figure 6.11: The Pixel Plot[1] (Section 5.5). A simulated 'true' MSA using INDELible is compared with MSA's generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), and ProPIP with k=1 under STFT with fiter: welch and filter size: 128 (MSA 5). Note: black pixel represents Characters and white pixel represents Indels.

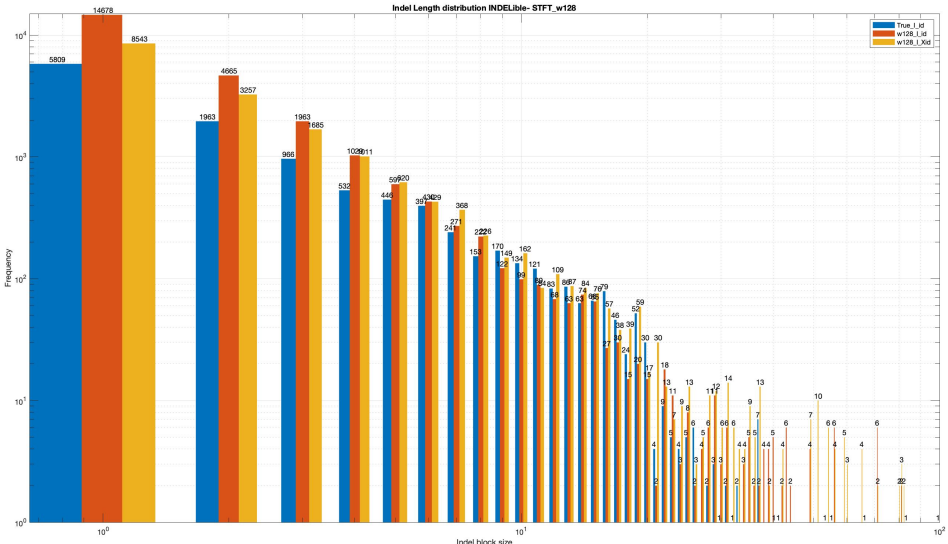


Figure 6.12: The Log-Log Plot. The 'true' Indel length distribution of INDELible data is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by ProPIP with k=1 under STFT with fiter: welch and filter size: 128. Note: The legend blue represents 'true' Indel length distribution, red represents inferred indel length distribution and yellow represents the inferred indel block distribution.

6.1.7 Quality check using qscore

Using the traditional MSA quality scoring tool named qscore [19], we observed the quality of all the observed alignments. We produced different quality scores provided by this tool and created the box plots of different scores computed for every tool used in this study (including the additional features of ProPIP).

To begin with, we received the best quality score in PRANK followed by MAFFT. Among the four scores provided by qscore we consider mainly the Q score or SPS score and Modeler score [same definition as explained in Section 3.6.3) to make conclusions. As described in the Section 3.6.3 the SPS and Modeler score differ only in their scaling term or denominator term. In Q score the denominator is the aligned pairs in reference or 'true' MSA whereas in Modeler score the scaling term is the aligned pairs in observed or inferred MSA. To add up, we observed similar pattern in the box plot of Q score and Modeler score, this endorses the fact that the aligned pairs are alike between true and test MSA's. On the one hand, the quality scores for PRANK is far better than our tool ProPIP and MAFFT. But, when we compare MAFFT and our tool, the quality scores end up almost equal with relatively smaller difference. This score of ProPIP is further improved slightly when the k-Factor method is applied, especially when k=0.25 and k=0.10.

To conclude, the quality test using qscore produced supporting and encouraging results in favor of our tool. We have witnessed improvement in our aligner using the k-Factor method, which suggest that our tool is slightly closer to PRANK in terms of quality scores (Q score and Modeler score). However, since the studies about k-Factor is an ongoing research, more results are indispensable to make conclusions based on this new feature of our aligner.

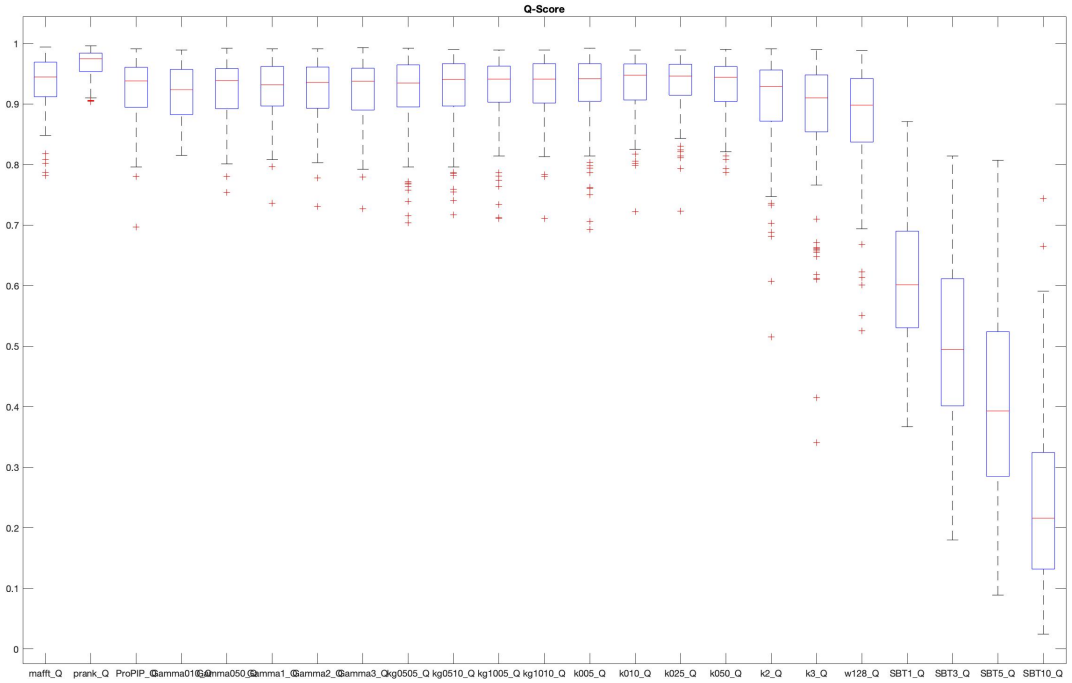


Figure 6.13: Box Plots of Q Scores for INDELible data. Reading from left Q Score box: Box 1: MAFFT, Box 2: PRANK, Box 3: ProPIP, Box 4: ProPIP with k=1 under Gamma n=4 and α=0.10, Box 5: ProPIP with k=1 under Gamma n=4 and α=0.50, Box 6: ProPIP with k=1 under Gamma n=4 and α=1, Box 7: ProPIP with k=1 under Gamma n=4 and α=2, Box 8: ProPIP with k=1 under Gamma n=4 and α=3, Box 9: ProPIP with k=0.05 under Gamma n=4 and α=0.05, Box 10: ProPIP with k=0.05 under Gamma n=4 and α=0.10, Box 11: ProPIP with k=0.10 under Gamma n=4 and α=0.05, Box 12: ProPIP with k=0.10 under Gamma n=4 and α=0.10, Box 13: ProPIP with k=0.05, Box 14: ProPIP with k=0.10, Box 15: ProPIP with k=0.25, Box 16: ProPIP with k=0.50, Box 17: ProPIP with k=2, Box 18: ProPIP with k=3, Box 19: ProPIP under STFT with FIR filter and filter size= 129. Box 20: ProPIP with k=1 under SBDP with T= 1, Box 21: ProPIP with k=1 under SBDP with T= 3, Box 22: ProPIP with k=1 under SBDP with T= 5, Box 23: ProPIP with k=1 under SBDP with T= 10.

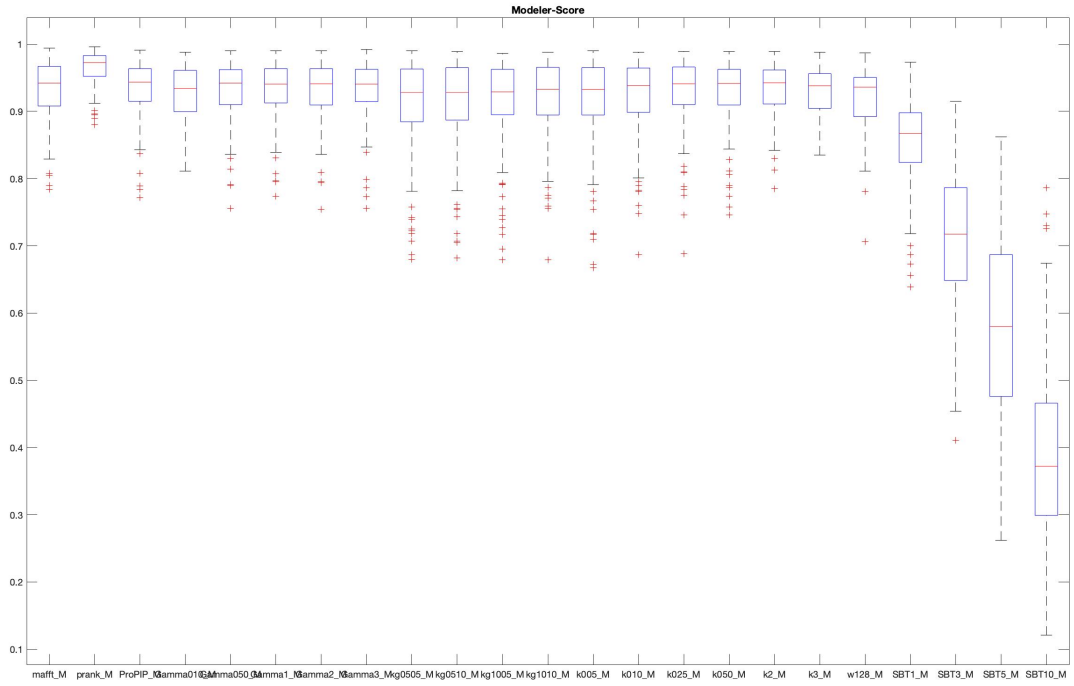


Figure 6.14: Box Plots of Modeler Scores for INDELible data. Reading from left Modeler Score box: Box 1: MAFFT, Box 2: PRANK, Box 3: ProPIP, Box 4: ProPIP with k=1 under Gamma n=4 and α=0.10, Box 5: ProPIP with k=1 under Gamma n=4 and α=0.50, Box 6: ProPIP with k=1 under Gamma n=4 and α=1, Box 7: ProPIP with k=1 under Gamma n=4 and α=2, Box 8: ProPIP with k=1 under Gamma n=4 and α=3, Box 9: ProPIP with k=0.05 under Gamma n=4 and α=0.05, Box 10: ProPIP with k=0.05 under Gamma n=4 and α=0.10, Box 11: ProPIP with k=0.10 under Gamma n=4 and α=0.05, Box 12: ProPIP with k=0.10 under Gamma n=4 and α=0.10, Box 13: ProPIP with k=0.05, Box 14: ProPIP with k=0.10, Box 15: ProPIP with k=0.25, Box 16: ProPIP with k=0.50, Box 17: ProPIP with k=2, Box 18: ProPIP with k=3, Box 19: ProPIP under STFT with FIR filter and filter size= 129. Box 20: ProPIP with k=1 under SBDP with T= 1, Box 21: ProPIP with k=1 under SBDP with T= 3, Box 22: ProPIP with k=1 under SBDP with T= 5, Box 23: ProPIP with k=1 under SBDP with T= 10.

6.2 PIP data analysis

6.2.1 Dynamics of indels in MAFFT, PRANK and ProPIP

The performance of different aligners such as; MAFFT, PRANK and ProPIP using PIP data is explained under this section using a summary statistics table (Table 6.8), a log-log plot (Figure 6.16), and a Pixel plot (Figure 6.15).

Firstly, the Table 6.8 shows the comparative summary statistics of the 'true' PIP alignments and inferred MSA's. Overall, the ProPIP obtained a closer fit to the 'true' indel length distribution. To start with, we compared the three aligners using the number of indels, maximum indel length and the mean value of the indel length distribution generated from each of the aligners. The total number of indels present in the 'true' alignments was 11648 with a maximum indel length of 4 and mean value of 1.082. The traditional aligner MAFFT underestimated the number of events occurring and over-aligned the sequences, which resulted a smaller number of indels on each alignment (in total 8996 indels) with maximum indel length of 5 and an average indel length of 1.095. Whereas PRANK inferred more than 10000 indels in total, with same maximum indel length and mean value as that of MAFFT. Comparatively, our aligner ProPIP inferred almost same (differ by a small number) statistics as that of 'true', as expected[15]. In other words, we can say that we actually tested the efficiency of our tool on the single character indel model PIP (same as the model used for data generation). Based on the log-log graph (Figure 6.16) of this comparative study, we observed a closer fit of all indel lengths for our aligner. We further tested the Indel block method explained in Section 5.3. Since we expect more single indel events under this datatype (PIP), we check the size of both Xid and id indel lists. If they both are having same number or closer number of indel which says that the indels placed by our aligner is having less number of indel events which are disturbed with single indel characters placed in between long indels. Therefore, based on the indel length distribution we could say that our aligner ProPIP is preferred more over other traditional aligners when used under single character indel model. Furthermore, we visualized a sample MSA from the set of alignments generated using the tool PIPJava with the help of Pixel-plot from SuiteMSA software. In the example we used, all the aligners obtained similar MSA length as that of 'true' alignment, differ by very small value. Compared to PRANK and MAFFT our tool attained similar indel placements with 'true', especially the head and tail region of the MSA that we used. In sum, we found that in contrast to other traditional aligners our aligner ProPIP infers relatively same number of indels as that of 'true'. This also means that ProPIP is more reliable when we use it under single character indel model. Similar to the framework used in the INDELible data analysis, we further studied the dynamics of indels using the additional features offered by ProPIP.

(100,8)	True (id)	MAFFT v7.453		PRANK v.170427		ProPIP	
		id	Xid	id	Xid	id	Xid
nIndels	11648	8996	8803	10058	9585	11023	10295
Max-IL	4	5	5	5	5	4	5
Mean	1.082	1.095	1.119	1.124	1.179	1.058	1.133
Median	1	1	1	1	1	1	1
SD	0.296	0.348	0.386	0.417	0.494	0.244	0.389

Table 6.8: The summary statistics of the 'true' Indel length distribution of PIP data (True(id)) is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by MAFFT v7.453, PRANK v.17042, ProPIP. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

Indel-length	True (count)	default ProPIP	Discrete Gamma		k-Factor	
			α=0.10	α=3	k=0.05	k=3
1	10756	10410	10713	10391	9533	11260
2	830	588	616	595	513	695
3	58	24	43	23	25	44
4	4	1	3	1	1	2

Table 6.9: Indel length frequency variation in 'true' PIP data, default ProPIP, ProPIP under Discrete Gamma distribution and ProPIP under k-Factor.

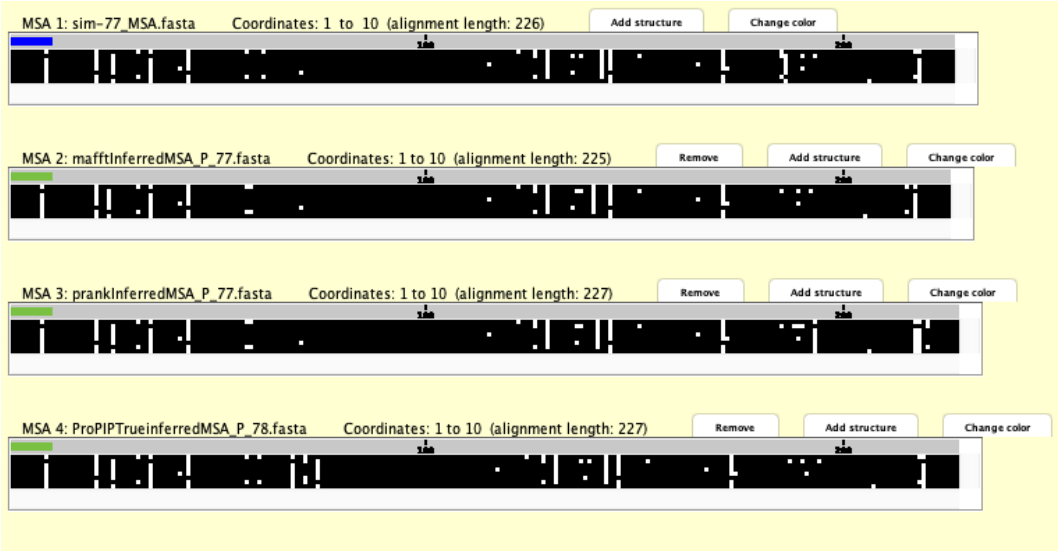


Figure 6.15: The Pixel Plot[1] (Section 5.5). A simulated 'true' MSA using PIPJava is compared with MSA's generated by MAFFT v7.453 (MSA 2), PRANK (MSA 3), and ProPIP (MSA 4). Note: black pixel represents Characters and white pixel represents Indels.

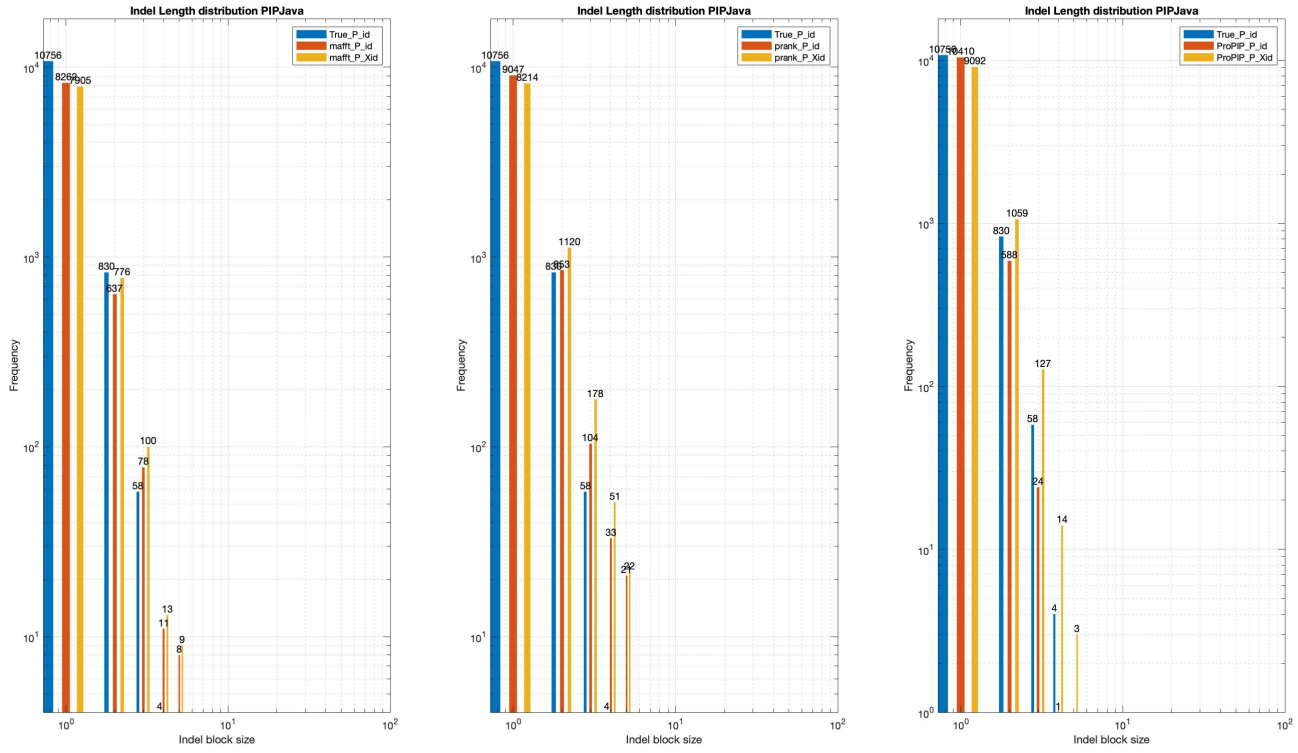


Figure 6.16: The Log-Log Plot. The 'true' Indel length distribution of PIP data is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by MAFFT v7.453, PRANK v.17042, ProPIP. Note: The legend blue represents 'true' Indel length distribution, red represents inferred indel length distribution and yellow represents the inferred indel block distribution.

6.2.2 Dynamics of indels in ProPIP under Discrete Gamma distribution

Using the feature Discrete Gamma distribution provided by ProPIP we investigated the pattern of indels in the inferred MSAs. The results obtained using this method is described in this section using a summary statistics table (Table 6.10), an indel length distribution graph (Figure 6.18) and an MSA visualization tool, SuiteMSA (6.17).

The Discrete Gamma distribution method produced same number of indels (vary in small number) as that of default ProPIP and with a slight improvement in the frequency of short indels. To elaborate, we tested the alignments inferred by this feature. For doing this we followed the framework explained in Section 5.1.4. As explained in the last section, the default ProPIP inferred 11023 indels with maximum indel length of 4 (same as in 'true' indel length distribution) and an average indel length of 1.058. The frequency of indel lengths occurring in the 'true', default ProPIP and ProPIP under Discrete Gamma distribution alignments is given in Table 6.9. In this table the difference in the frequency of indel lengths (in ProPIP under Discrete Gamma distribution) are observed to be more fit to 'true' when we used a relatively small α value except for indel length of 2. The possible reason could be the forced insertion of single characters in between gaps. Interestingly, the indel length statistics obtained when α= 3 resembles the default ProPIP results to a greater extent. Moreover, we investigated the indel placements using Pixel-plot which again confirms these observations which we have inferred from the summary statistics table and log-log plot. Also, for the particular reference MSA we used in the Pixel-plot our tool inferred exactly same reference MSA length when α= 0.10, 0.50 and 1 (MSA 5 to MSA 7 in Figure 6.17). To conclude, we obtained positive results using ProPIP under Discrete Gamma distribution, which helps to tune the default ProPIP output and to create a closer fit to 'true' indel length distribution.

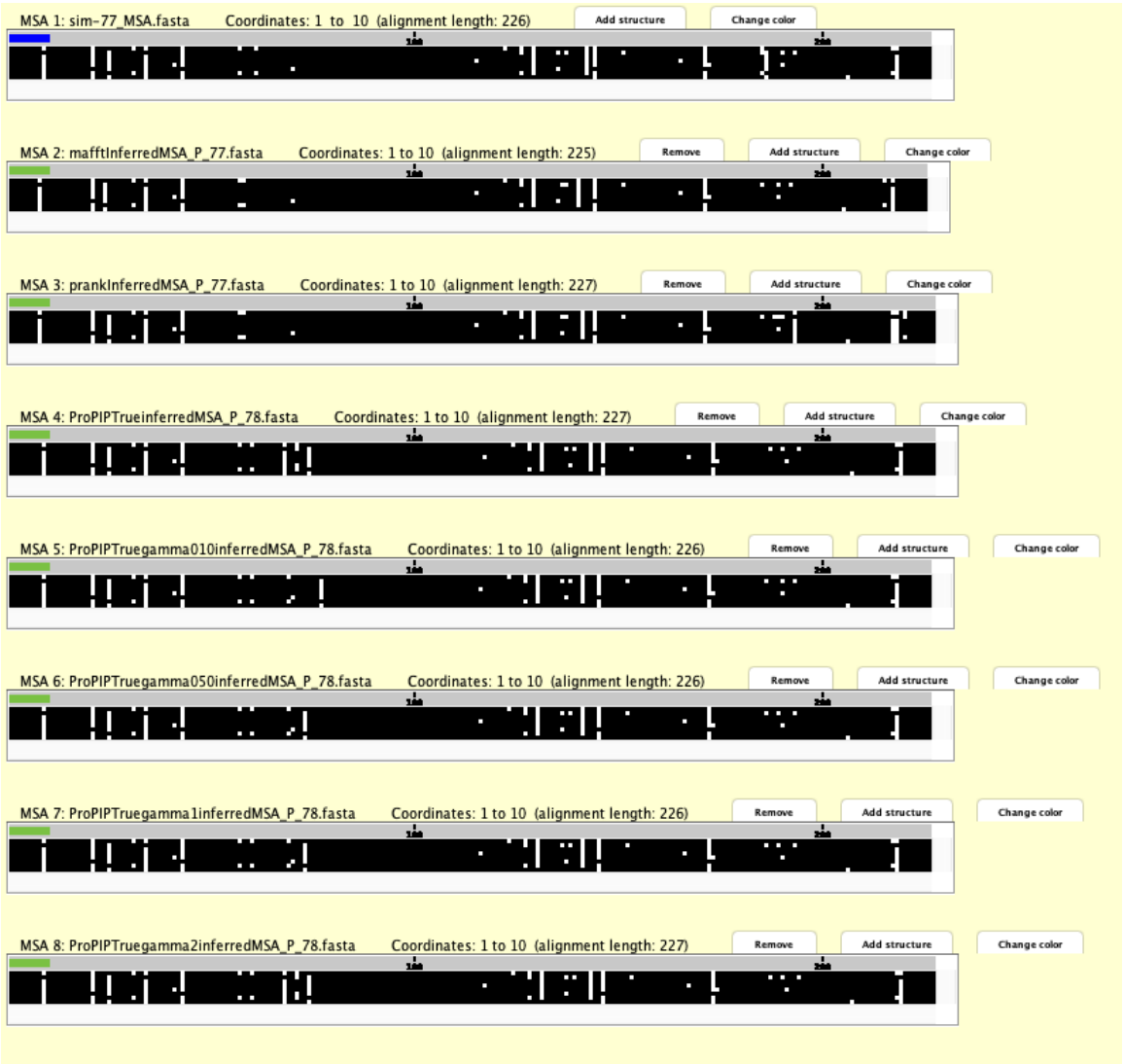


Figure 6.17: The Pixel Plot[1] (Section 5.5). A simulated 'true' MSA using PIPJava is compared with MSA's generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), ProPIP with k=1 under Gamma n=4, α=0.10 (MSA 5), ProPIP with k=1 under Gamma n=4, α=0.50 (MSA 6), ProPIP with k=1 under Gamma n=4, α=1 (MSA 7), and ProPIP with k=1 under Gamma n=4, α=2 (MSA 8). Note: black pixel represents Characters and white pixel represents Indels.

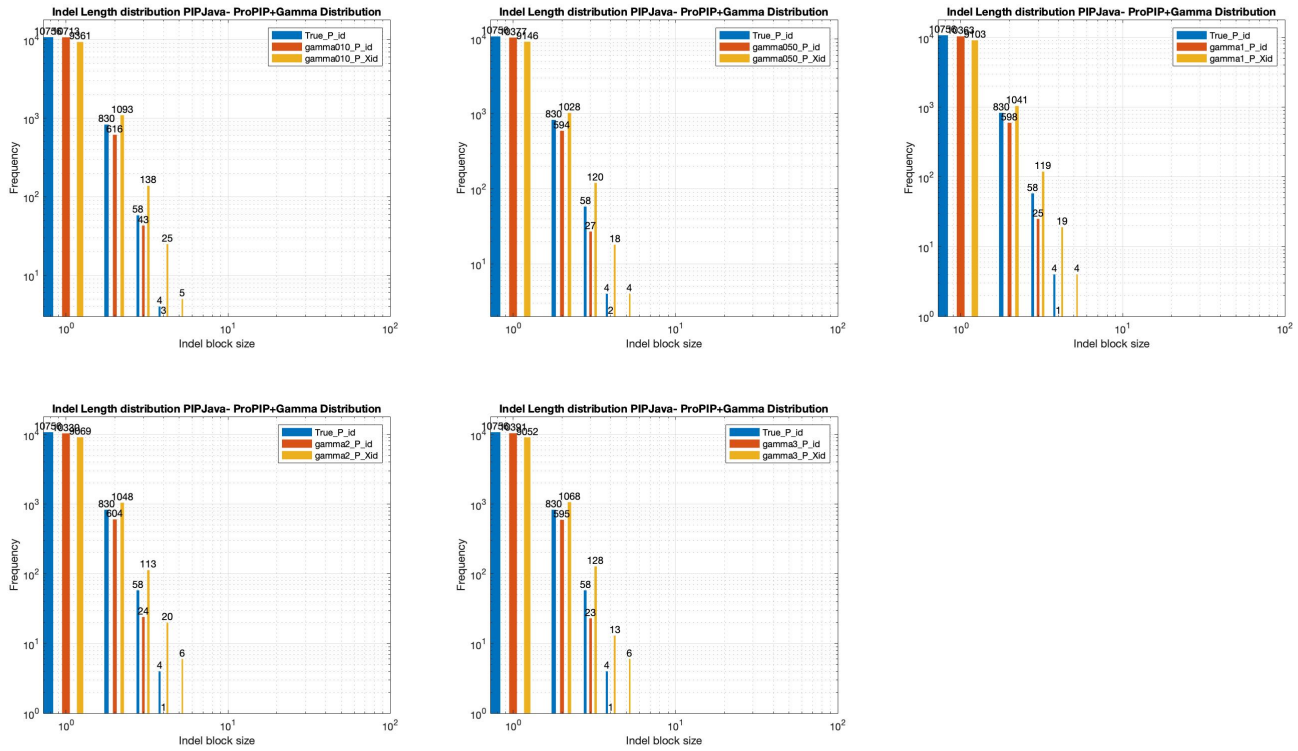


Figure 6.18: The Log-Log Plot. The 'true' Indel length distribution of PIP data is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by ProPIP with k=1 under Gamma n=4, α=0.10, ProPIP with k=1 under Gamma n=4, α=0.50 , ProPIP with k=1 under Gamma n=4, α=1, ProPIP with k=1 under Gamma n=4, α=2, and ProPIP with k=1 under Gamma n=4, α=3. Note: The legend blue represents 'true' Indel length distribution, red represents inferred indel length distribution and yellow represents the inferred indel block distribution.

	True	G0.10		G0.50		G1		G2		G3	
(100,8)	(id)	id	Xid	id	Xid	id	Xid	id	Xid	id	Xid
nIndels	11648	11375	10622	11000	10316	10987	10286	10959	10256	11010	10267
Max-IL	4	4	5	4	5	4	5	4	5	4	5
Mean	1.082	1.062	1.138	1.060	1.130	1.059	1.131	1.060	1.132	1.0585	1.1351
Median	1	1	1	1	1	1	1	1	1	1	1
SD	0.296	0.260	0.406	0.249	0.389	0.247	0.391	0.247	0.395	0.245	0.395

Table 6.10: The summary statistics of the 'true' Indel length distribution of PIP data (True(id)) is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by ProPIP with k=1 under Gamma n=4, α=0.10, ProPIP with k=1 under Gamma n=4, α=0.50 , ProPIP with k=1 under Gamma n=4, α=1, ProPIP with k=1 under Gamma n=4, α=2, ProPIP with k=1 under Gamma n=4, α=3. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

6.2.3 Dynamics of indels in ProPIP under k-Factor

With the help of the new method introduced in ProPIP called k-Factor we tested the possibility of tuning the ProPIP alignments inferred from PIP data. The results obtained using this method is described in this section using the summary statistics table (Table 6.11), log-log graph (Figure 6.20) and the Pixel-plot (6.19).

More supporting and motivating results for the new method k-Factor is obtained from this study using PIP data. In detail, based on the experiment setup explained in the Section 5.1.5 we tested the dynamics of indels in the MSA's reconstructed by ProPIP under different k values. The Table 6.9 provides the frequency of indel lengths occurring in the 'true', default ProPIP and ProPIP under k-Factor alignments. On the one hand, we observed a gradual increase in the number of indels when we increased the value of k above 1. For instance, when k is 3 the ProPIP

overestimated the number of events and produced a total number of 12001 indels, which is slightly greater than the true number of indels (11648). However, in this same value of k our tool obtained a similar fit as that of the result which we obtained when we experimented the ProPIP under Discrete Gamma distribution using α=0.10. On the other hand, when we decreased the value of k below 1 we observed that the ProPIP is underestimating the number of events and produced the lowest number of indel events (10072), which is comparatively much lower than the default ProPIP results and very close to the number of indels estimated by PRANK. Furthermore, we observed that the variation of indel lengths in this method is similar to that of Discrete Gamma distribution method. More precisely, when we decreased the value of α (below 1) we obtained the same statistics as that of increasing the value of k (above 1). Therefore, we can relate them as identical to each other i.e., when k is higher the among sites rate variation is also higher or α is relatively smaller. As this is an ongoing research and out of this report, we cannot provide more information on this cross relation. Moreover, we manually examined and compared the true and inferred MSAs side by side to check for indel patterns using the Pixel-plot. We observed that the indel patterns and MSA lengths are changing at very few places for the particular reference alignment that we used. In conclusion, we received more meaningful results using the k-Factor method for the ongoing research and also for this study using PIP data.

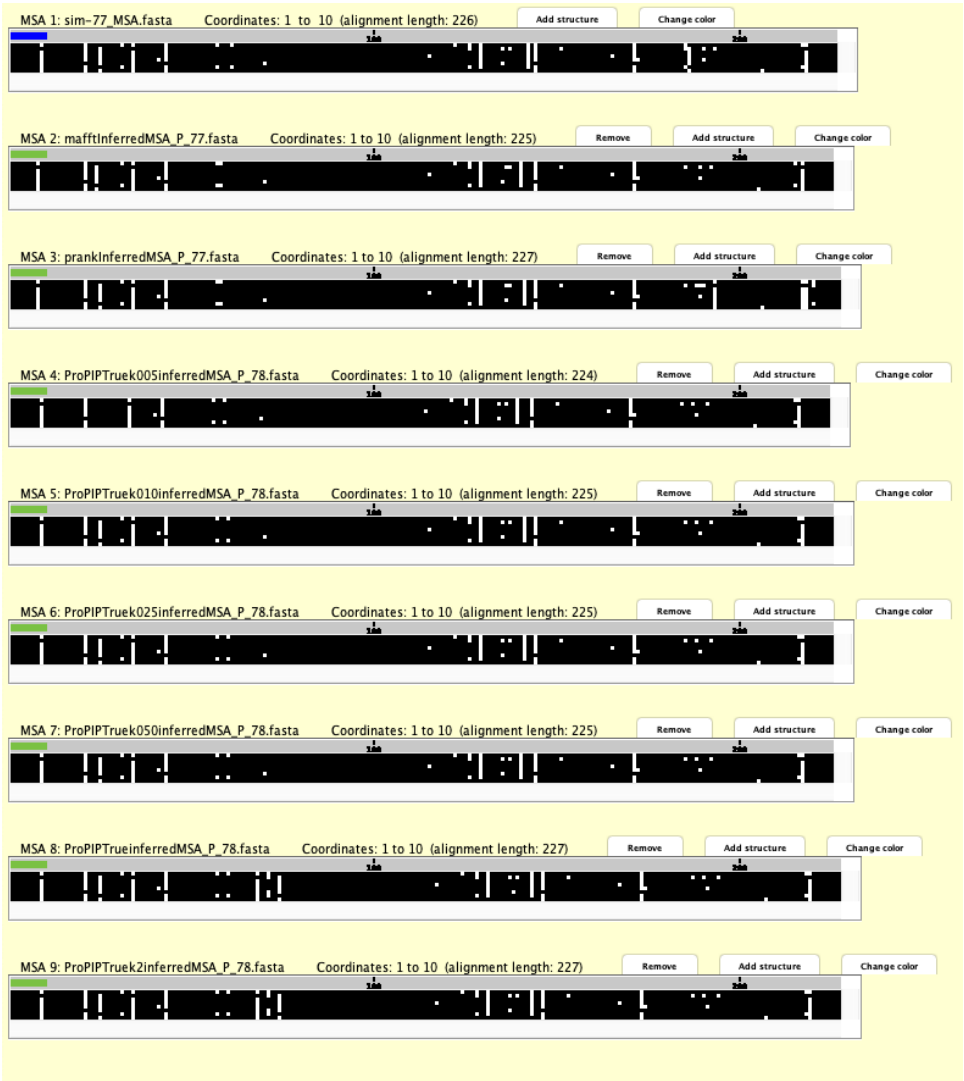


Figure 6.19: The Pixel Plot[1] (Section 5.5). A simulated 'true' MSA using PIPJava is compared with MSA's generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=0.05 (MSA 4), ProPIP with k=0.10 (MSA 5), ProPIP with k=0.25 (MSA 6), ProPIP with k=0.50 (MSA 7), ProPIP with k=1 (MSA 8), and ProPIP with k=2 (MSA 9). Note: black pixel represents Characters and white pixel represents Indels.

(100,8)	True (id)	k0.05		k0.10		k0.25		k0.50		k2		k3	
		id	Xid	id	Xid	id	Xid	id	Xid	id	Xid	id	Xid
nIndels	11648	10072	9530	10199	9652	10513	9899	10692	100053	11444	10682	12001	11093
Max-IL	4	4	4	4	5	4	5	4	5	4	5	4	5
Mean	1.082	1.056	1.116	1.058	1.118	1.058	1.123	1.058	1.126	1.062	1.137	1.066	1.153
Median	1	1	1	1	1	1	1	1	1	1	1	1	1
SD	0.296	0.242	0.357	0.245	0.357	0.244	0.377	0.247	0.378	0.252	0.400	0.264	0.422

Table 6.11: The summary statistics of the 'true' Indel length distribution of PIP data (True(id)) is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by ProPIP with k=0.05, ProPIP with k=0.10, ProPIP with k=0.25, ProPIP with k=0.50, ProPIP with k=2, ProPIP with k=3. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

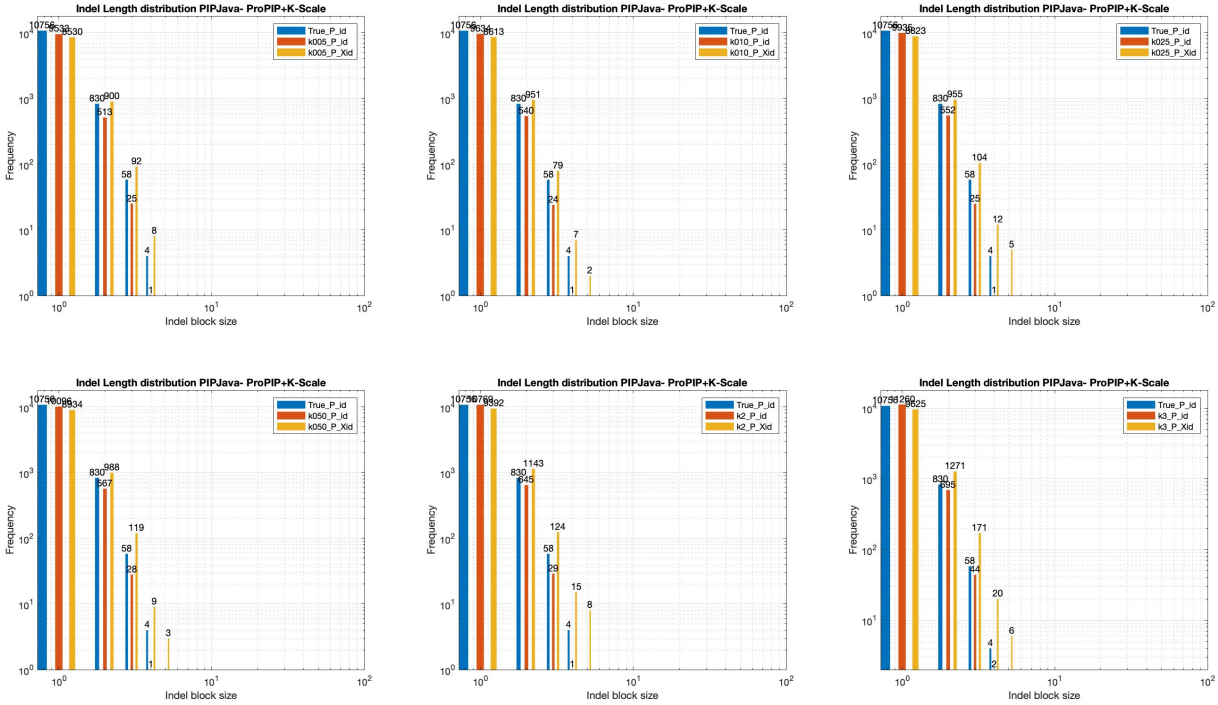


Figure 6.20: The Log-Log Plot. The 'true' Indel length distribution of PIP data is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by (from top-left to bottom-right) ProPIP with k=0.05, ProPIP with k=0.10, ProPIP with k=0.25, ProPIP with k=0.50, ProPIP with k=2, and ProPIP with k=3. Note: The legend blue represents 'true' Indel length distribution, red represents inferred indel length distribution and yellow represents the inferred indel block distribution.

6.2.4 Dynamics of indels in ProPIP under Discrete Gamma distribution and k-Factor

The combined effect of Discrete Gamma distribution and k-Factor method is experimented using PIP data to understand the indel dynamics. This section elaborates the results obtained from this study using a statistics table (Table 6.12), log-log graph (Figure 6.22) and the Pixel-plot (6.21).

Overall, we obtained some interesting results from this experiment. To begin with, we reconstructed the alignments from input multiple sequences using ProPIP under k-Factor and Discrete Gamma distribution methods at the same time. As explained earlier, the 'true' indel length distribution contains 11648 indels with a maximum indel length of 4 and mean value of 1.082. Also, the default ProPIP inferred 11023 indels with maximum indel length of 4 (same as in 'true' indel length distribution) and an average indel length of 1.058. Compared to these results our tool under this method estimated lower number (10106) of indel events, which is very close to the number

of indels inferred by PRANK (from Table 6.12). The plausible reason for this could be the low value of k that we used in the alignment setup of this experiment. In Figure 6.22, we observed for all scenarios under this experiment that the frequency of indel lengths are maintaining the similar counts (differ in small values). This could be due to the smaller values of α and k that we selcted in this experiment. Due to time constraints and presence of same interesting results from k-Factor and Discrete Gamma distribution when applied individually, we tested the other supplementary features of ProPIP using the same data. We further visualized the same reference MSA that we used in the previous section using Pixel-plot and compared the inferred MSA's side by side. The only change that we observed in this particular alignment under this plot is the indel placements at the middle region of the reconstructed MSAs. For instance, when the value of k and α is 0.05 (MSA 5 in Figure 6.21) we witnessed similar indel placements in the inferred MSA to 'true'. In sum, the experiment to tune PIP data using ProPIP under Discrete Gamma distribution and k-Factor together at the same time produced some valuable results for this study and for the future works.

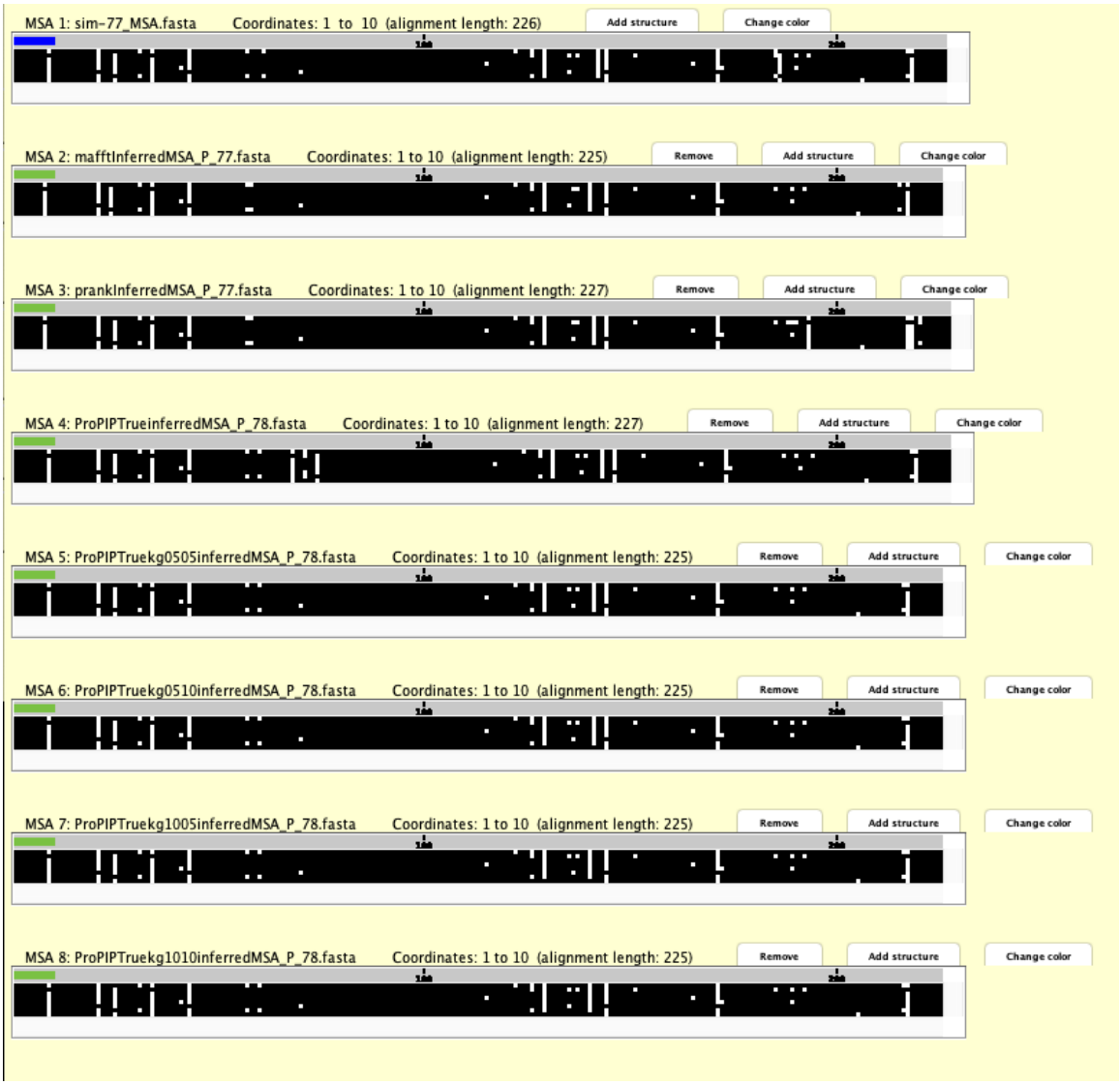


Figure 6.21: The Pixel Plot[1] (Section 5.5). A simulated 'true' MSA using PIPJava is compared with MSA's generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), ProPIP with k=0.05 under Gamma n=4, α=0.05 (MSA 5), ProPIP with k=0.05 under Gamma n=4, α=0.10 (MSA 6), ProPIP with k=0.10 under Gamma n=4, α=0.05 (MSA 7), and ProPIP with k=0.10 under Gamma n=4, α=0.10 (MSA 8), Note: black pixel represents Characters and white pixel represents Indels.

(100,8)	True (id)	kg0505		kg0510		kg1005		kg1010	
		id	Xid	id	Xid	id	Xid	id	Xid
nIndels	11648	10106	9546	10152	9584	10270	9706	10272	9688
Max-IL	4	4	5	4	5	4	5	4	5
Mean	1.082	1.058	1.120	1.057	1.120	1.062	1.123	1.060	1.121
Median	1	1	1	1	1	1	1	1	1
SD	0.296	0.248	0.369	0.246	0.368	0.255	0.371	0.247	0.374

Table 6.12: The summary statistics of the 'true' Indel length distribution of PIP data (True(id)) is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by ProPIP with k=0.05 under Gamma n=4, α=0.05, ProPIP with k=0.05 under Gamma n=4, α=0.10, ProPIP with k=0.10 under Gamma n=4, α=0.05, ProPIP with k=0.10 under Gamma n=4, α=0.10. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

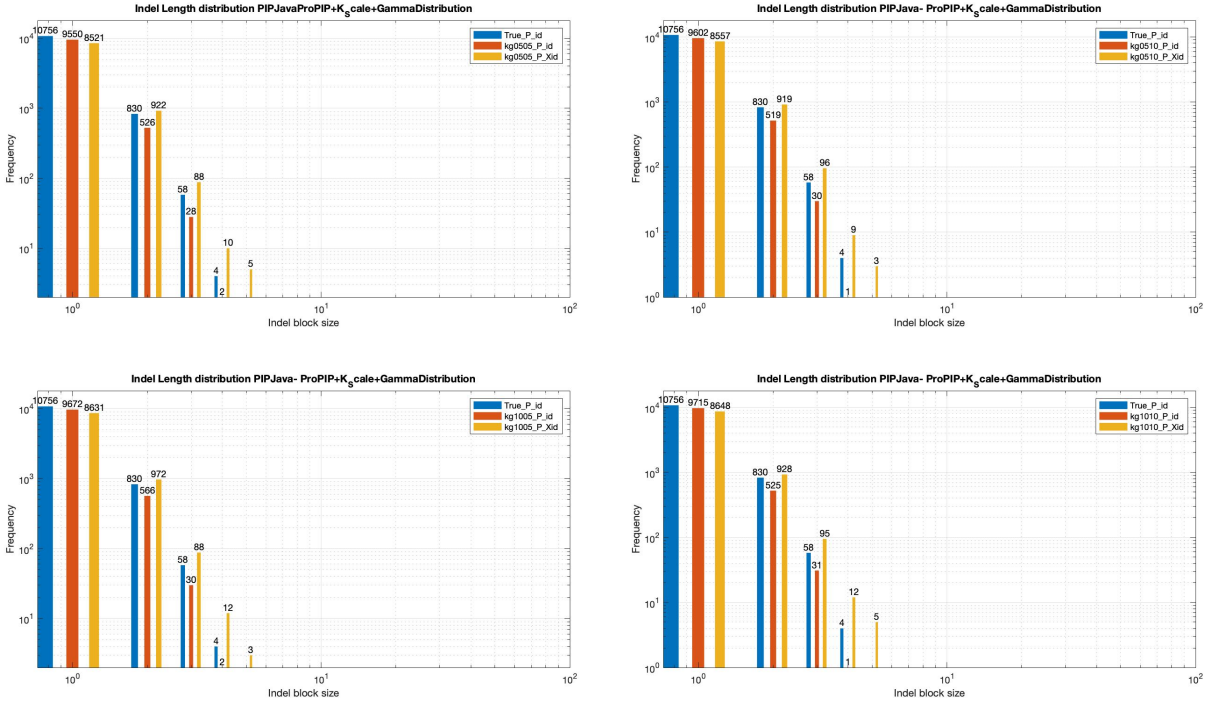


Figure 6.22: The Log-Log Plot. The 'true' Indel length distribution of PIP data is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by (from top-left to bottom-right) ProPIP with k=0.05 under Gamma n=4, α=0.05, ProPIP with k=0.05 under Gamma n=4, α=0.10, ProPIP with k=0.10 under Gamma n=4, α=0.05, and ProPIP with k=0.10 under Gamma n=4, α=0.10. Note: The legend blue represents 'true' Indel length distribution, red represents inferred indel length distribution and yellow represents the inferred indel block distribution.

6.2.5 Dynamics of indels in ProPIP under Stochastic Backtracking

The results obtained for the experiments conducted using ProPIP under Stochastic Backtracking DP algorithm is explained in this Section. To explain the results we use a summary statistics table (Table 6.13), a log-log graph (Figure 6.23) and the Pixel-plot (6.24).

First of all, the results obtained using this feature does not provide any useful insights for this work towards understanding the indel dynamics. To explain, as per the framework explained in Section 5.1.7 we have run the distortion parameter T with the PIP data. We received extremely different statistical values for this experiment compared to the 'true' indel statistics. For instance, one of the trends we noticed in this study is the rapid increase in the number of indels (from 63185 to 85451) as we decrease the T value (from 10 to 1). One possible reason could be the greedy nature of the dynamic programming used in this algorithm. More interestingly, the value of maximum indel length is also following a similar trend, that is when the value of T is reduced from 10 to 1, the size of the indels inferred in the alignments is also decreasing faster (from 370

to 25). We further studied this experiment results with the help of a log-log graph to understand the frequency distribution of the indel lengths. We observed a high boost in the frequency of indel lengths in all scenarios tested under this study. Furthermore, we compared the inferred alignments from this experiment side by side using the same reference MSA used in the previous section. From the Pixel-plot we visually observed the presence of huge single character insertions in the MSA's inferred by ProPIP under SBDP.

In sum, we have learned many things from the MSA's inferred using ProPIP under SBDP for our future work, even though the results obtained using SBDP does not looks closer to the reference alignments. Also, this study helped us to understand the tuning mechanism of the distortion parameter used in this algorithm.

(100,8)	True (id)	T1		T3		T5		T10	
		id	Xid	id	Xid	id	Xid	id	Xid
nIndels	11648	85451	36277	75142	29963	68682	26905	63185	24162
Max-IL	4	25	81	60	113	168	168	370	410
Mean	1.082	2.303	5.425	2.756	6.911	3.016	7.700	3.153	8.245
Median	1	2	4	2	4	2	4	2	4
SD	0.296	1.782	5.600	2.588	8.975	3.213	10.843	4.785	14.121

Table 6.13: The summary statistics of the 'true' Indel length distribution of PIP data (True(id)) is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by ProPIP with k=1 under SBDP with T= 1, ProPIP with k=1 under SBDP with T= 3, ProPIP with k=1 under SBDP with T= 5, ProPIP with k=1 under SBDP with T= 10. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

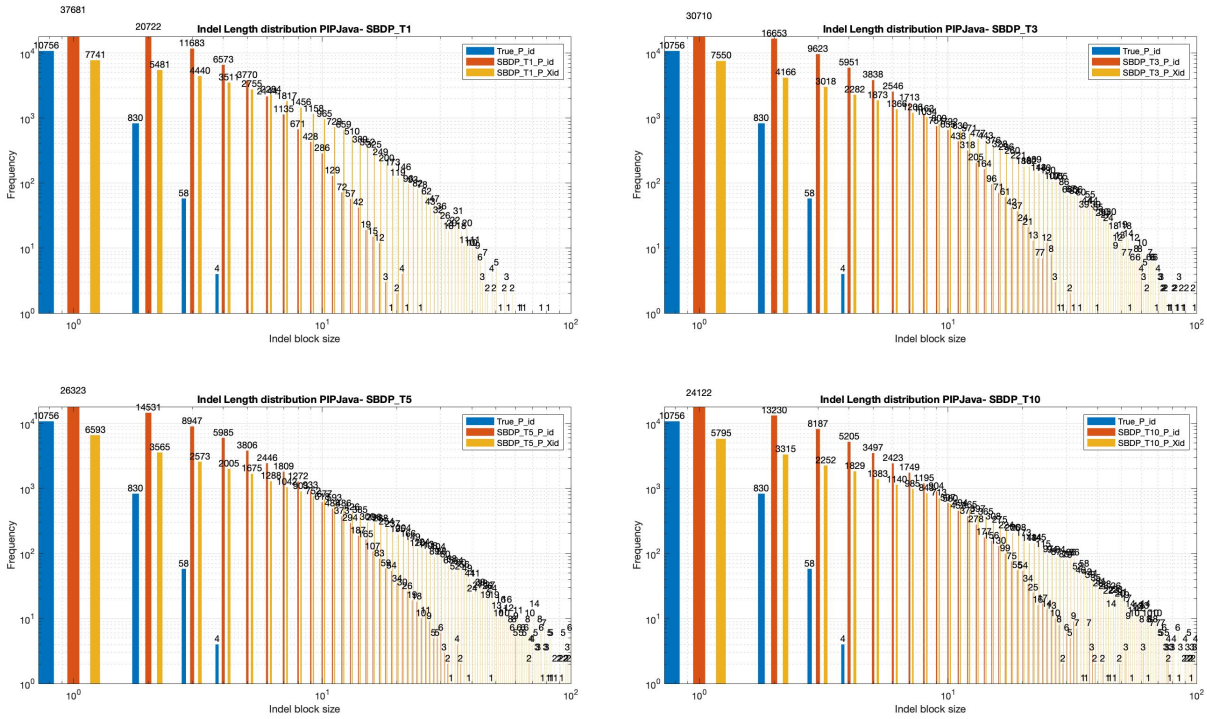


Figure 6.23: The Log-Log Plot. The 'true' Indel length distribution of PIP data is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by (from top-left to bottom-right) ProPIP with k=1 under SBDP with T= 1, ProPIP with k=1 under SBDP with T= 3, ProPIP with k=1 under SBDP with T= 5, and ProPIP with k=1 under SBDP with T= 10. Note: The legend blue represents 'true' Indel length distribution, red represents inferred indel length distribution and yellow represents the inferred indel block distribution.

Figure 6.24: The Pixel Plot[1] (Section 5.5). A simulated 'true' MSA using PIPJava is compared with MSA's generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), ProPIP with k=1 under SBDP with T= 1 (MSA 5), ProPIP with k=1 under SBDP with T= 3 (MSA 6), ProPIP with k=1 under SBDP with T= 5 (MSA 7), and ProPIP with k=1 under SBDP with T= 10 (MSA 8). Note: black pixel represents Characters and white pixel represents Indels.

6.2.6 Dynamics of indels in ProPIP under Short Time Fourier Transform

This section explains the results obtained from the Short-Time Fourier Transform method implemented in ProPIP. We have tested this feature using the *welch* window function with window size of 128. We explain the results with the help of a statistical table (Table 6.14), log-log plot (Figure 6.26) and the Pixel-plot (Figure 6.25).

Overall, the STFT feature does not produced useful insights as expected using the configurations explained in Section 5.1.8. In detail, we have reconstructed the MSA's using STFT method and collected the indel length information from the MSA's using the methods explained in Section 5.2 and Section 5.3. Using this information of indel lengths we have calculated the corresponding summary statistics and plotted the indel length distribution graphs in log-log scale. The frequency of short indel lengths obtained for this dataset fits with 'true' short indel lengths of size 1 and 2, even though the indel length information includes large set of long indel lengths with a maximum indel length of 111. Interestingly, the number of indels obtained is more closer to the 'true' number of indel events. The probable reason could be the large standard deviation that we have obtained from the inferred indel length information ('true' SD= 0.296 and STFT SD= 3.246). More interestingly, the frequency of short indels of length 1,2 and 3 in this method seems to be similar as that of the same in PRANK. However, to understand more about this indel placement one has to look closer into the algorithms implemented in this feature. Moreover, we compared the reconstructed MSA's with a reference alignment (same alignment that we used in the previous section) from this dataset using the Pixel-plot. The only conceivable insight we have obtained from the MSA is the closer MSA length (differ in small value) that we obtained for this particular inferred MSA.

To conclude, the experiment using STFT produced more distinct frequency distribution of indel lengths to 'true' and default ProPIP, even though some regions in the indel length distribution fits the 'true' frequency distribution of indel lengths.

(100,8)	True (id)	w128	
		id	Xid
nIndels	11648	11853	10768
Max-IL	4	111	131
Mean	1.082	1.359	1.495
Median	1	1	1
SD	0.296	3.246	4.361

Table 6.14: The summary statistics of the 'true' Indel length distribution of PIP data (True(id)) is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by ProPIP with k=1 under STFT with fiter: welch and filter size: 128. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

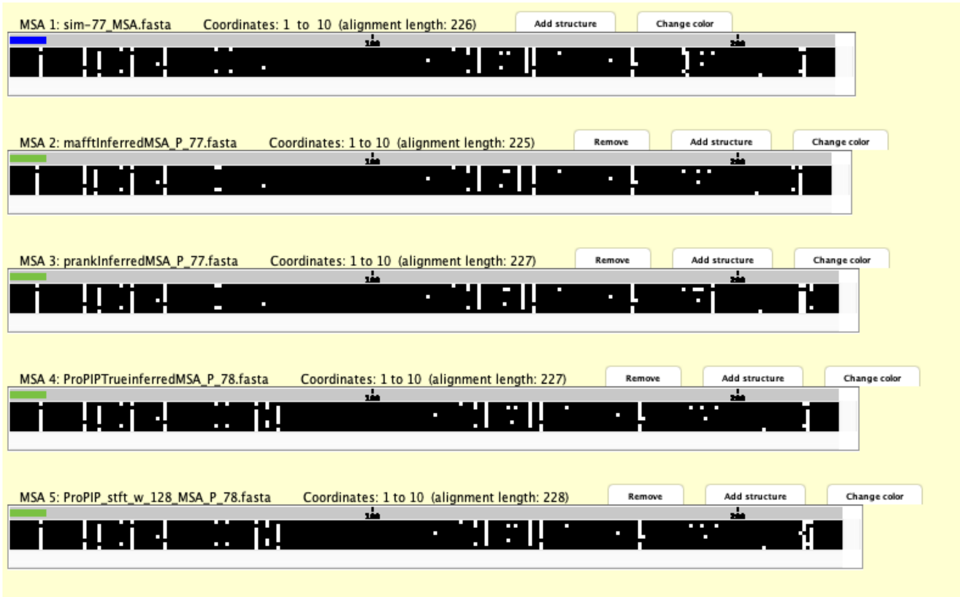


Figure 6.25: The Pixel Plot[1] (Section 5.5). A simulated 'true' MSA using PIPJava is compared with MSA's generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), and ProPIP with k=1 under STFT with fiter: welch and filter size: 128 (MSA 5). Note: black pixel represents Characters and white pixel represents Indels.

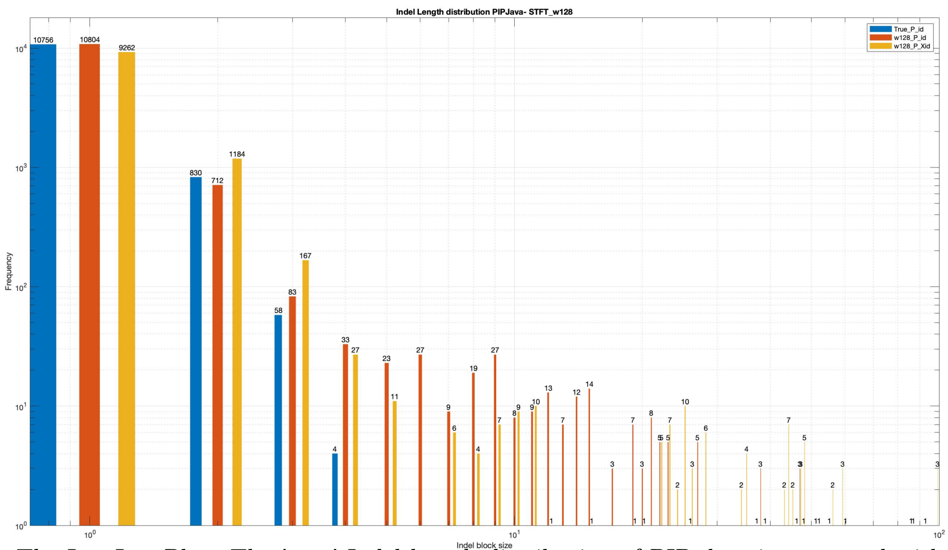


Figure 6.26: The Log-Log Plot. The 'true' Indel length distribution of PIP data is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by ProPIP with k=1 under STFT with fiter: welch and filter size: 128. Note: The legend blue represents 'true' Indel length distribution, red represents inferred indel length distribution and yellow represents the inferred indel block distribution.

6.2.7 Quality check using qscore

To analyze the quality of the inferred MSA's we produced different quality scores provided by the standard multiple sequence alignment quality scoring program, qscore. To explain we plotted the box plots of the scores (Figure 6.27 and Figure 6.28) that we calculated using this tool under PIP data.

Overall, our tool achieved the maximum Q score (differ in tiny Q score value) compared to other aligners that we used in this study. In detail, we used mainly the Q score and Modeler score to make conclusions from this experiment. The comparison of quality scores obtained for ProPIP, PRANK and MAFFT resulted in favor of our tool followed by other two aligners. Since Q and Modeler results graph are identical it can be derived that the aligned pairs are identical between reference and test MSA's (except for SBDP results). The experiment under the Discrete Gamma distribution (α=3) and the k-Factor method with k=0.05 produced equal Q score values which confirms the findings explained in their respective result sections. In sum, we received encouraging and motivating results which support our tool and its additional features, which is helpful for the future works and the ongoing research.

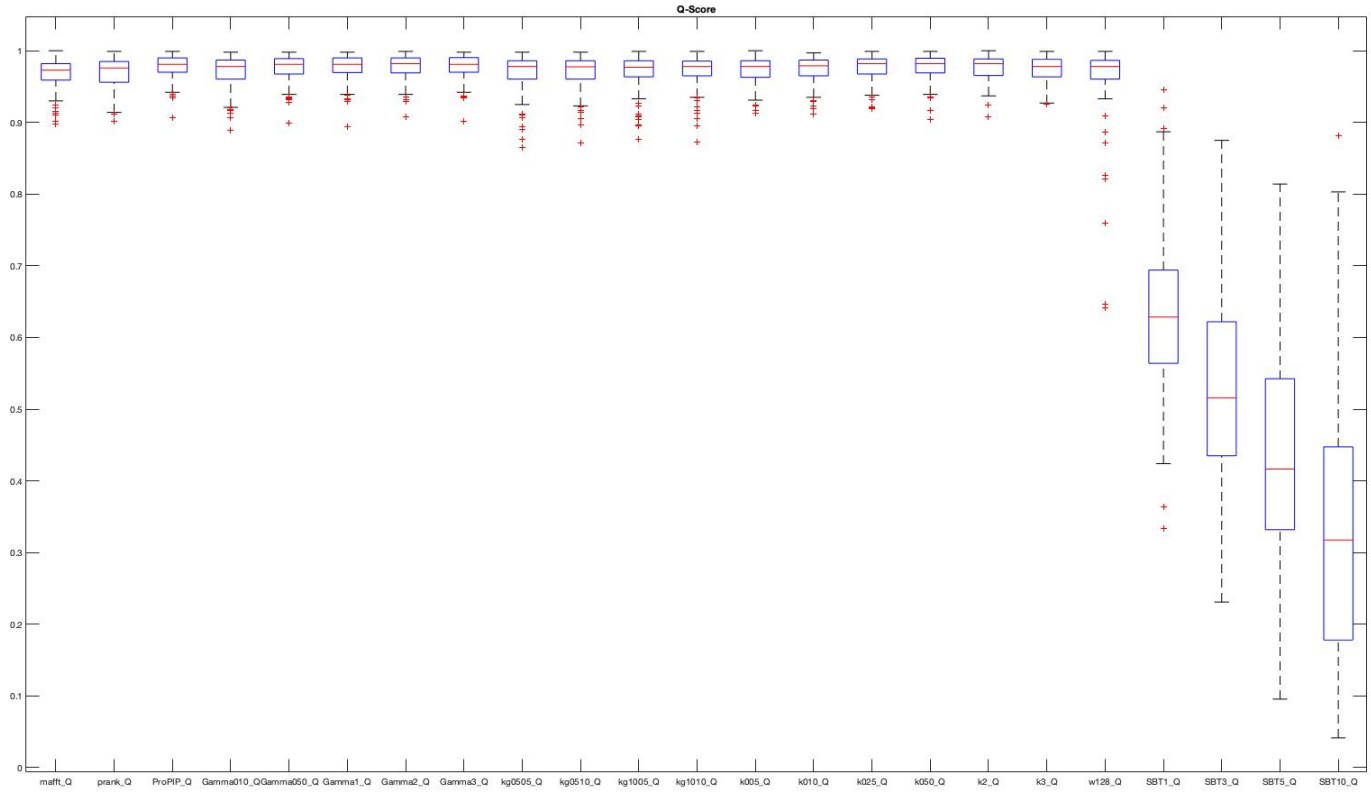


Figure 6.27: Box Plots of Q Scores for PIP data. Reading from left Q Score box: Box 1: MAFFT, Box 2: PRANK, Box 3: ProPIP, Box 4: ProPIP with k=1 under Gamma n=4 and α=0.10, Box 5: ProPIP with k=1 under Gamma n=4 and α=0.50, Box 6: ProPIP with k=1 under Gamma n=4 and α=1, Box 7: ProPIP with k=1 under Gamma n=4 and α=2, Box 8: ProPIP with k=1 under Gamma n=4 and α=3, Box 9: ProPIP with k=0.05 under Gamma n=4 and α=0.05, Box 10: ProPIP with k=0.05 under Gamma n=4 and α=0.10, Box 11: ProPIP with k=0.10 under Gamma n=4 and α=0.05, Box 12: ProPIP with k=0.10 under Gamma n=4 and α=0.10, Box 13: ProPIP with k=0.05, Box 14: ProPIP with k=0.10, Box 15: ProPIP with k=0.25, Box 16: ProPIP with k=0.50, Box 17: ProPIP with k=2, Box 18: ProPIP with k=3, Box 19: ProPIP under STFT with FIR filter and filter size= 129. Box 20: ProPIP with k=1 under SBDP with T= 1, Box 21: ProPIP with k=1 under SBDP with T= 3, Box 22: ProPIP with k=1 under SBDP with T= 5, Box 23: ProPIP with k=1 under SBDP with T= 10.

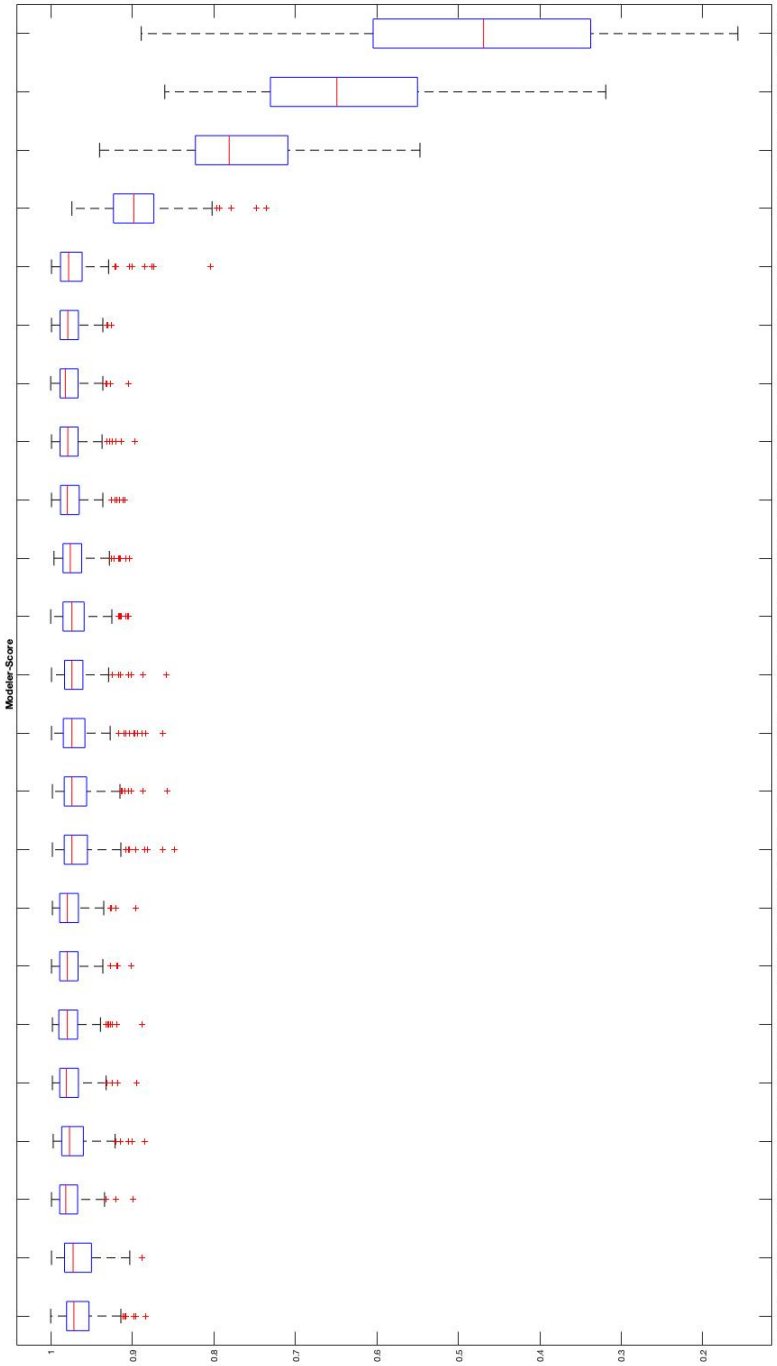


Figure 6.28: Box Plots of Modeler Scores for PIP data. Reading from left Modeler Score box: Box 1: MAFFT, Box 2: PRANK, Box 3: ProPIP, Box 4: ProPIP with k=1 under Gamma n=4 and α=0.10, Box 5: ProPIP with k=1 under Gamma n=4 and α=0.50, Box 6: ProPIP with k=1 under Gamma n=4 and α=1, Box 7: ProPIP with k=1 under Gamma n=4 and α=2, Box 8: ProPIP with k=1 under Gamma n=4 and α=3, Box 9: ProPIP with k=0.05 under Gamma n=4 and α=0.05, Box 10: ProPIP with k=0.05 under Gamma n=4 and α=0.10, Box 11: ProPIP with k=0.10 under Gamma n=4 and α=0.05, Box 12: ProPIP with k=0.10 under Gamma n=4 and α=0.10, Box 13: ProPIP with k=0.05, Box 14: ProPIP with k=0.10, Box 15: ProPIP with k=0.25, Box 16: ProPIP with k=0.50, Box 17: ProPIP with k=2, Box 18: ProPIP with k=3, Box 19: ProPIP under STFT with FIR filter and filter size= 129. Box 20: ProPIP with k=1 under SBDP with T= 1, Box 21: ProPIP with k=1 under SBDP with T= 3, Box 22: ProPIP with k=1 under SBDP with T= 5, Box 23: ProPIP with k=1 under SBDP with T= 10.

6.3 Real data analysis

6.3.1 Dynamics of indels in MAFFT, PRANK and ProPIP

In this section we compare the alignments, inferred from the four 13 taxa protein sequences explained under the Section 4.3, by MAFFT, PRANK and our aligner ProPIP. The results obtained from this study are provided below in form of a summary statistics table (Table 6.15), indel length distribution plot (Figure 6.30), and a Pixel-plot (Figure 6.29) with a short description.

To begin with, the statistical table (Table 6.15) provides the number of indels, maximum indel length, mean, median and mode of the indel length distribution of the protein alignments and reconstructed MSA's. The total number of indels provided by the sequence alignments of protein was 314 with a maximum indel length of 43, mean value of 8.930 and SD equals to 11.517. Since this protein alignments were aligned using CLUSTALW 1.8 as explained in Section 4.3, we decided to use PRANK as the reference tool for this dataset in order to keep the same comparative study structure and also due to exceptionally accurate results of PRANK in previous studies [15] (applicable to all the following sections). Therefore, the new reference statistics (same as PRANK statistics) consists of 532 indels with a maximum indel length of 63, mean value of 8.325 and SD of 10.170. Based on this, from the Table 6.15, we could say that the MAFFT is underestimating the number of events and results smaller number of indel events (343) with a maximum indel size of 48, mean value of 8.481 and SD of 11.883. At the same time, our tool overestimated the number of events which lead to higher number of indels (1309) with maximum indel length of 34 and an average indel length of 2.828. From the indel block distribution statistics (Xid), we can say that our aligner ProPIP is splitting the longer indels in most of the instances (number of Xid indels is 866). Therefore one possible reason that we could point out here is the presence of large amount of short indels inferred by our tool. This observation is apparently visible from the log-log graph provided with the indel length distribution of sequence alignments of protein and inferred MSA's. When we compare the statistics values in Table 6.15, among the three aligners MAFFT and PRANK produced an indel length distribution which is closer to the 'real' indel distribution (note that we have no valid information about the 'real' data that we use). We further, visualized one of the protein alignments and the corresponding inferred MSA's using the Pixel-plot provided by the SuiteMSA software to understand the indel dynamics occurring in the alignments. From the Pixel plot (Figure 6.29), one of the observations that we obtained is the size similarity of the particular MSA that we used. That is, our tool and PRANK received almost similar MSA size (differ in small value), however, the indel placements in our MSA does not agree with the patterns inferred in PRANK and in the reference MSA that we used in this example.In sum, we found that our aligner ProPIP infers extremely large number of single indel events (699) than MAFFT (136) and PRANK (105). Therefore we decided to experiment the supplementary features of ProPIP to understand the indel dynamics and indel placements.

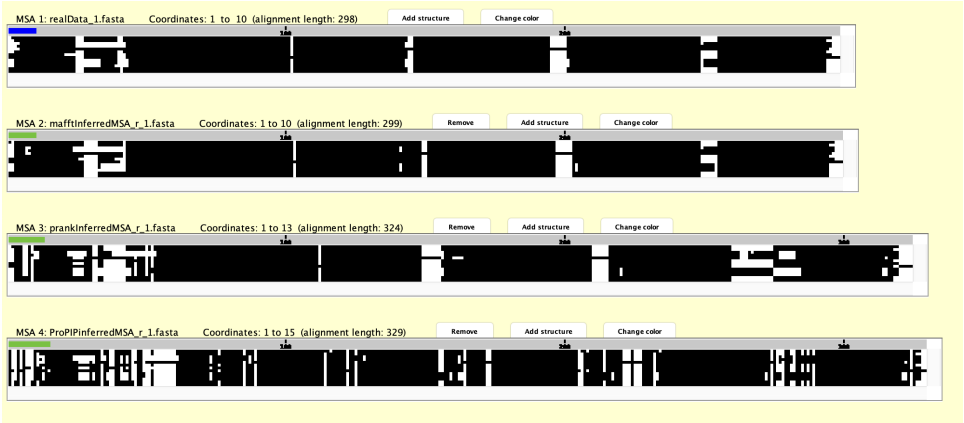


Figure 6.29: The Pixel Plot[1] (Section 5.5). A protein alignment (MSA 1 Prot_atpB) from the study is compared with MSA's generated by MAFFT v7.453 (MSA 2), PRANK (MSA 3), ProPIP (MSA 4). Note: black pixel represents Characters and white pixel represents Indels.

(4,13)	True(id)	MAFFT v7.453		PRANK v.170427		ProPIP	
		id	Xid	id	Xid	id	Xid
nIndels	314	343	338	532	475	1309	866
Max-IL	43	48	48	63	81	34	59
Mean	8.930	8.481	8.610	8.325	9.324	2.828	4.275
Median	4	3	3	5	6	1	2
SD	11.517	11.883	11.952	10.170	10.725	3.368	6.267

Table 6.15: The summary statistics of the indel length distribution of protein alignments (True(id)) used in the study is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by MAFFT v7.453, PRANK v.17042, ProPIP. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

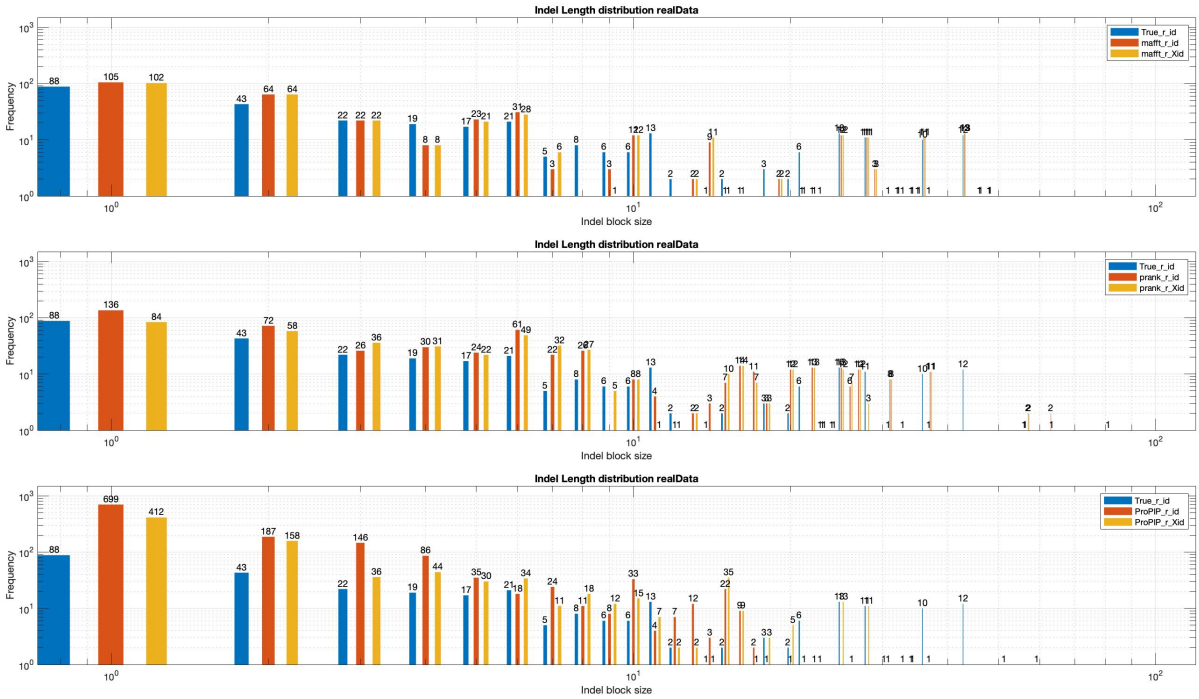


Figure 6.30: The Log-Log Plot. The indel length distribution of protein alignments (simulated using CLUSTALW1.8 (Ref: [13])) used in the study is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by MAFFT v7.453, PRANK v.17042, and ProPIP. Note: The legend blue represents protein alignment indel length distribution, red represents inferred indel length distribution and yellow represents inferred indel block distribution.

6.3.2 Dynamics of indels in ProPIP under Discrete Gamma distribution

The experimental results (based on the framework explained in Section 5.1.4) obtained for ProPIP under Discrete Gamma distribution is provided in this section. The results are represented using a summary statistics table (Table 6.16), a log-log graph (Figure 6.32), and a Pixel-plot (Figure 6.31).

Compared to the ProPIP results (Table 6.15) explained in the last section, the ProPIP under Discrete Gamma distribution method did not produced a higher variation in the number of indels and the maximum indel length. In detail, based on the experiment setup explained in the Section 5.1.4, we tried to tune the indel placements in the alignments. From the table 6.16, we have observed that the number of indels remains almost same when the MSA's are reconstructed using α= 2 and α= 3, where the amount of indels on each set of inferred MSA's corresponds to 1302 and 1306 respectively. Note that, as mentioned in the earlier section we used PRANK as the reference tool for this analysis. In this study, we observed a rise in the number of indels when we tuned the value of α near to zero (except for α=0.50). Based on the explanation of Discrete Gamma

distribution method explained in Section 5.1.4, when α is set to a value closer to zero the among site rate variation is higher, therefore we could say that the rise in the amount of indels is due to this rate variation on each site explained by this method. Also, from the log-log plot the variation of short indel events are visible, for instance, the number of single indel events are increasing when we tune down the α value to a relatively smaller value (except for α=0.50). In order to understand indel patterns in the protein alignments and inferred alignments we compared them side by side using an MSA visualization tool called Pixel-plot. In Figure 6.31, for this particular MSA, our aligner inferred an MSA of closer length as that of PRANK when α= 2. However, we witnessed no pattern in the MSA's inferred using this method in contrast to default ProPIP MSA's.

Figure 6.31: The Pixel Plot[1] (Section 5.5). A protein alignment reconstructed by PRANK v.170427 (MSA 1) is compared with MSA's generated by MAFFT v7.453 (MSA 2), ProPIP with k=1 (MSA 3), ProPIP with k=1 under Gamma n=4, α=0.10 (MSA 4), ProPIP with k=1 under Gamma n=4, α=0.50 (MSA 5), ProPIP with k=1 under Gamma n=4, α=1 (MSA 6), ProPIP with k=1 under Gamma n=4, α=2 (MSA 7), and ProPIP with k=1 under Gamma n=4, α=3 (MSA 8). Note: black pixel represents Characters and white pixel represents Indels.

	PRANK	G0.10		G0.50		G1		G2		G3	
(100,8)	**(id)**	id	Xid	id	Xid	id	Xid	id	Xid	id	Xid
nIndels	532	1450	966	1211	792	1311	841	1302	843	1306	845
Max-IL	63	34	43	34	56	33	51	33	51	34	52
Mean	8.325	2.4813	3.725	2.788	4.264	2.784	4.340	2.783	4.299	2.775	4.289
Median	5	1	1	1	1	1	2	1	1	1	1
SD	10.170	2.841	6.367	3.219	6.932	3.578	6.684	3.376	6.379	3.310	6.549

Table 6.16: The summary statistics of the indel length distribution of PRANK alignments (PRANK(id)) used in the study is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by ProPIP with k=1 under Gamma n=4, α=0.10, ProPIP with k=1 under Gamma n=4, α=0.50 , ProPIP with k=1 under Gamma n=4, α=1, ProPIP with k=1 under Gamma n=4, α=2, ProPIP with k=1 under Gamma n=4, α=3. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

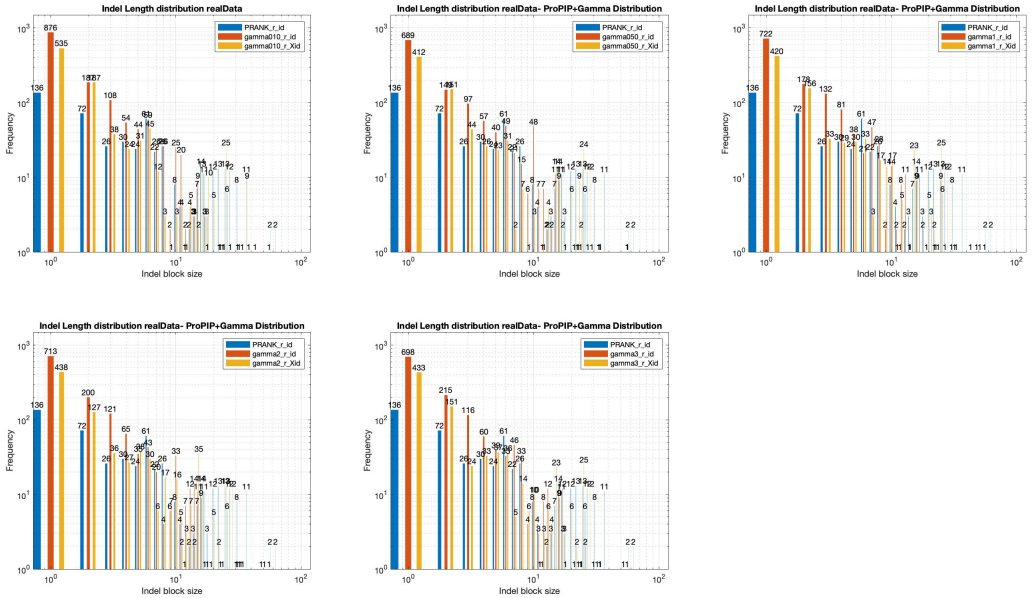


Figure 6.32: The Log-Log Plot. The indel length distribution of PRANK alignments used in the study is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by ProPIP with k=1 under Gamma n=4, α=0.10, ProPIP with k=1 under Gamma n=4, α=0.50 , ProPIP with k=1 under Gamma n=4, α=1, ProPIP with k=1 under Gamma n=4, α=2, and ProPIP with k=1 under Gamma n=4, α=3. Note: The legend blue represents PRANK indel length distribution, red represents inferred indel length distribution and yellow represents inferred indel block distribution.

6.3.3 Dynamics of indels in ProPIP under k-Factor

The new feature introduced through this study, k-Factor is applied to ProPIP to reconstruct the sequence of protein alignments under the experimental settings explained in the Section 5.1.5. We elaborate the results using a summary table (Table 6.17), a log-log graph (Figure 6.34), and the Pixel-plot (Figure 6.33).

Overall, the new method showed us encouraging results to vary the amount of indels when an MSA is reconstructed from multiple protein sequences. Using the default ProPIP settings our tool inferred 1309 indels with maximum indel length of 34. However, using k-Factor method we were able to reduce the number of indels to a minimum of 968 and maximum indel length of 33 with a k value of 0.05. In detail, we could say that the gradual decrease in the number of indels in the inferred MSAs is when we scale the ProPIP parameter (λ and μ) values using a relatively smaller value of k (here we used 0.05). The possible reason for this could be the decrease in the number of single indel events. This observation could be explained using the log-log graph, as we can see from the graph (Figure 6.34), the frequency of single indel event is at its peak value 1464 when k has a maximum value of 3 and the occurrence of single indel events is at its minimum (968) when the parameter k has a low value of 0.05. Thus, we could say that the variation in the number of indels is highly dependent on the value of k. Furthermore, we also analysed the inferred MSA's in a single window using the Pixel plot tool. In the Pixel plot, we compared the alignments inferred using different k values to inferred MSA's from the other two state of art aligners PRANK and MAFFT. For this particular MSA, our aligner under relatively smaller k value obtained a closer number of columns as that of MAFFT, differ by a small length.

	PRANK	k0.05		k0.10		k0.25		k0.50		k2		k3	
(4,13)	**(id)**	id	Xid	id	Xid	id	Xid	id	Xid	id	Xid	id	Xid
nIndels	532	968	625	993	653	1087	721	1277	865	1898	1156	2329	1466
Max-IL	63	33	55	33	55	34	56	34	59	35	66	34	51
Mean	8.325	2.320	3.594	2.324	3.520	2.209	3.332	2.044	3.017	2.409	3.956	2.204	3.501
Median	5	1	2	1	2	1	2	1	1	1	1	1	1
SD	10.170	2.874	5.523	2.725	6.082	2.902	5.729	2.337	4.947	2.915	6.036	2.558	5.835

Table 6.17: The summary statistics of the indel length distribution of PRANK alignments (PRANK(id)) used in the study is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by ProPIP with k=0.05, ProPIP with k=0.10, ProPIP with k=0.25, ProPIP with k=0.50, ProPIP with k=2, ProPIP with k=3. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

Figure 6.33: The Pixel Plot[1] (Section 5.5). A protein alignment reconstructed by PRANK v.170427 (MSA 1) is compared with MSA's generated by MAFFT v7.453 (MSA 2), ProPIP with k=0.05 (MSA 3), ProPIP with k=0.10 (MSA 4), ProPIP with k=0.25 (MSA 5), ProPIP with k=0.50 (MSA 6), ProPIP with k=1 (MSA 7), and ProPIP with k=2 (MSA 8). Note: black pixel represents Characters and white pixel represents Indels.

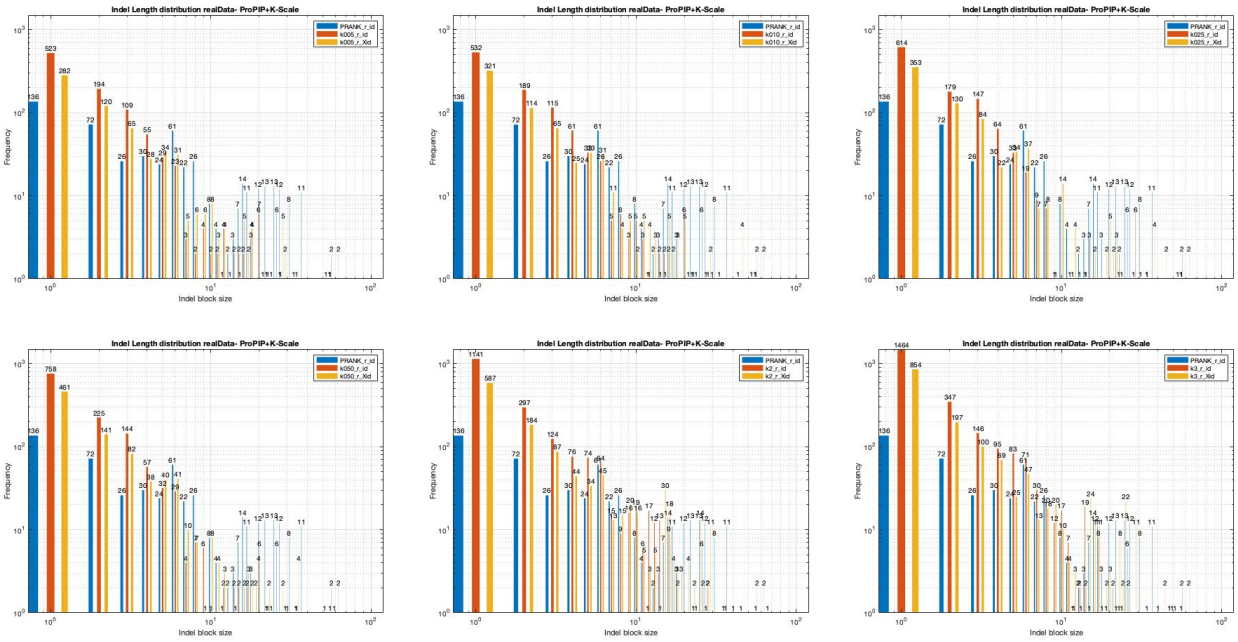


Figure 6.34: The Log-Log Plot. The indel length distribution of PRANK alignments used in the study is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by (from top-left to bottom-right) ProPIP with k=0.05, ProPIP with k=0.10, ProPIP with k=0.25, ProPIP with k=0.50, ProPIP with k=2, and ProPIP with k=3. Note: The legend blue represents PRANK indel length distribution, red represents inferred indel length distribution and yellow represents inferred indel block distribution.

6.3.4 Dynamics of indels in ProPIP under Discrete Gamma distribution and k-Factor

We applied the Discrete Gamma distribution and k-Factor method at the same time in our tool, ProPIP to reconstruct the protein alignments. This section explains the results obtained from this study and are provided below in form of a summary statistics table (Table 6.18), indel length distribution plot (Figure 6.35), and a Pixel-plot (Figure 6.36) with a short description.

We obtained the lowest number of indels (827) with very low number of single indel events (447) in the reconstructed protein alignments under this experiment. We conducted this experiment based on the study design explained in the Section 5.1.6. We wanted to observe the performance of this experiment to default ProPIP and to other aligners. Based on the summary table we could say that this method is generating lower number of indel events compared to the amount of indels inferred by default ProPIP (in Table 6.15). We could say that this could be due to the presence of small value of k and α, which indeed controls the rate of insertion (λ) and deletion (μ) in our tool. However, the relation between α and k is still unclear and an ongoing study. Furthermore, we also analyzed the inferred MSA's in a single window using the Pixel plot tool. In the Pixel plot we compared the alignments inferred under this experiment to an MSA inferred by PRANK (as reference alignment) and MAFFT. For this particular MSA, our tool ProPIP under this method obtained a closer MSA length (301) as that of MAFFT (299), differ by a small length. Another change that we noticed in these alignments (MSA 4 to MSA 7) is the tidiness of the reconstructed MSA compared to default ProPIP (MSA 3) alignment. Especially, the indels present at the middle region of the alignment from default ProPIP is replaced to the head region of the MSA.

In sum, we observed the possibilities to adjust the number of indels in sequence of protein alignments by applying parameters from Discrete Gamma distribution and k-Factor at the same time. Since this is an ongoing research, providing more thorough results are out of this research work.

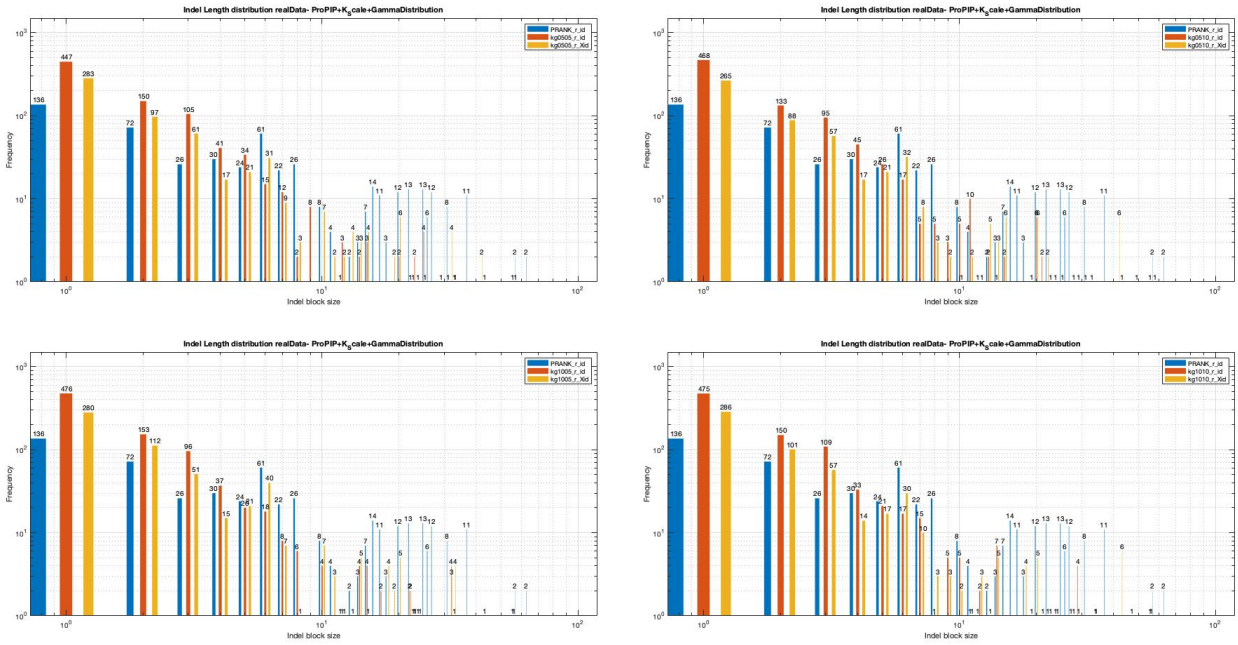


Figure 6.35: The Log-Log Plot. The indel length distribution of PRANK alignments used in the study is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by (from top-left to bottom-right) ProPIP with k=0.05 under Gamma n=4, α=0.05, ProPIP with k=0.05 under Gamma n=4, α=0.10, ProPIP with k=0.10 under Gamma n=4, α=0.05, and ProPIP with k=0.10 under Gamma n=4, α=0.10. Note: The legend blue represents PRANK indel length distribution, red represents inferred indel length distribution and yellow represents inferred indel block distribution.

(100,8)	PRANK (id)	kg0505		kg0510		kg1005		kg1010	
		id	Xid	id	Xid	id	Xid	id	Xid
nIndels	532	831	565	827	532	837	569	847	557
Max-IL	63	33	57	33	58	33	56	34	56
Mean	8.325	2.389	3.515	2.402	3.733	2.342	3.445	2.314	3.519
Median	5	1	1	1	2	1	2	1	1
SD	10.170	3.026	5.958	3.058	6.727	3.347	5.650	3.070	6.458

Table 6.18: The summary statistics of the indel length distribution of PRANK alignments (PRANK(id)) used in the study is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by ProPIP with k=0.05 under Gamma n=4, α=0.05, ProPIP with k=0.05 under Gamma n=4, α=0.10, ProPIP with k=0.10 under Gamma n=4, α=0.05, ProPIP with k=0.10 under Gamma n=4, α=0.10. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

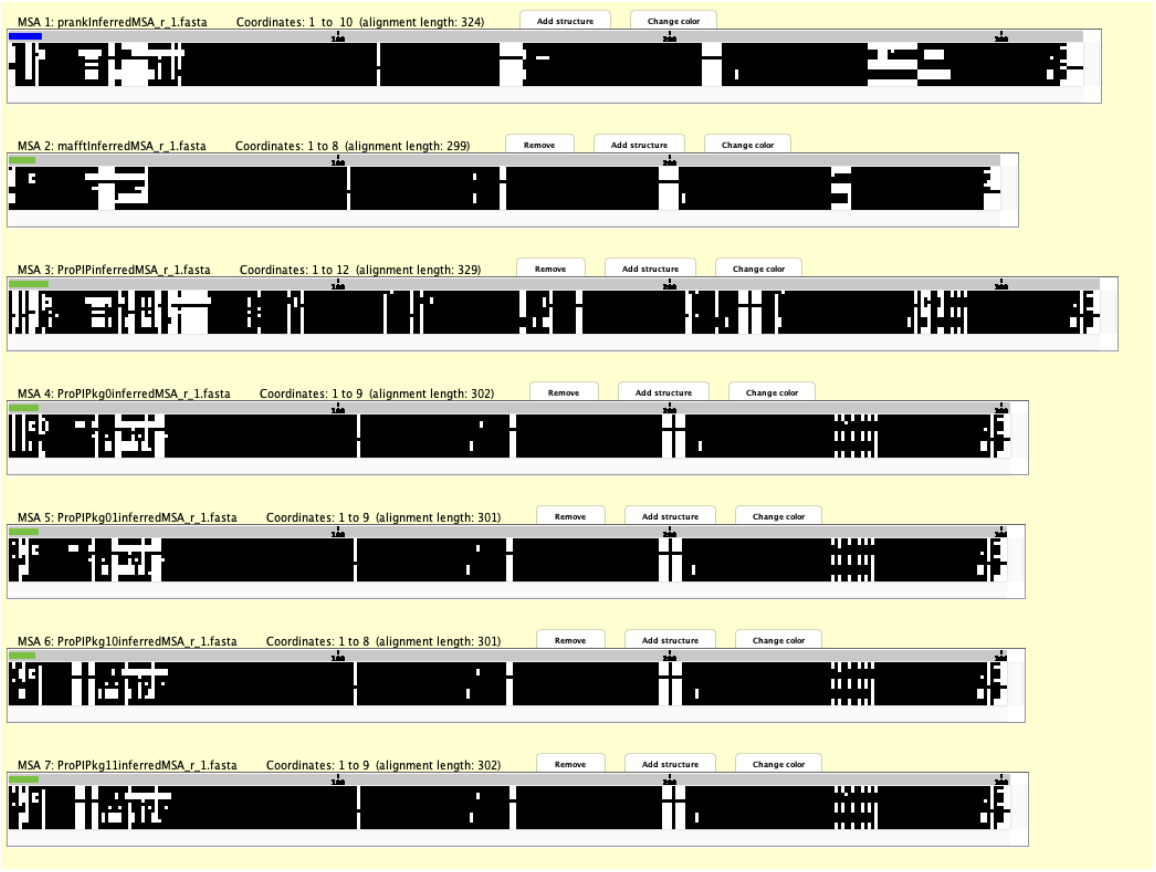


Figure 6.36: The Pixel Plot[1] (Section 5.5). A protein alignment reconstructed by PRANK v.170427 (MSA 1) is compared with MSA's generated by MAFFT v7.453 (MSA 2), ProPIP with k=1 (MSA 3), ProPIP with k=0.05 under Gamma n=4, α=0.05 (MSA 4), ProPIP with k=0.05 under Gamma n=4, α=0.10 (MSA 5), ProPIP with k=0.10 under Gamma n=4, α=0.05 (MSA 6), and ProPIP with k=0.10 under Gamma n=4, α=0.10 (MSA 7), Note: black pixel represents Characters and white pixel represents Indels.

6.3.5 Dynamics of indels in ProPIP under Short Time Fourier Transform

This section explains the results obtained using Short-Time Fourier Transform method (**??**) implemented in ProPIP. We have tested this feature using the *welch* window function with window size equal to 128. Based on the statistical table (Table 6.14), log-log plot (Figure 6.26) and the Pixel-plot (Figure 6.25) we observed less useful findings for this study.

The result of this experiment do not comply with our expectations. In detail, we tested the STFT method using the *welch* window function and window size equal to 128. As explained

earlier, the reference indel length distribution (same as PRANK indel length distribution) contains 532 indel events with a maximum indel length of 63 and mean value of 8.325. In contrast, the default ProPIP inferred 1309 indel events with maximum indel length of 34 and an average indel length of 3.368. Whereas ProPIP under STFT produced a greater number of (1991) indel events with almost same maximum indel length (44). Based on the number of indels present in the reconstructed protein alignments and the log-log indel length distribution plot we could say that the STFT method is overestimating the number of events and frequency of all the variants of indel lengths. For instance, we observed that the number of single indel events incremented from 699 (in default ProPIP) to 1097 (ProPIP under STFT) and this trend of increase is true for all the other indel lengths. However, interestingly, the only positive factor we received is the maximum indel length that is inferred by this method (maximum indel lendth of 44). We further wanted to see the MSA reconstructed by this method, for this we used the Pixel-plot tool from SuiteMSA. From Figure 6.25, for this particular protein alignment, we found that the MSA reconstructed by STFT is creating more number of indel events at the head and tail portion of the alignment compared to default ProPIP alignment. Also, we obtained a larger MSA length (360) as that of reference MSA (PRANK MSA of length 324). In sum, we tested the possibility to tune the indel placements when an MSA is reconstructed. However, based on this current study using *welch* filter with filter size 128 we have seen that it did not make any improvements over the default ProPIP. But there are other window functions implemented in this method, which we haven't tested due to time constraints. Therefore, a further study is necessary to make conclusive results on this method.

(4,13)	**PRANK (id)**	**w128**	
		id	Xid
nIndels	532	1991	1237
Max-IL	63	44	83
Mean	8.325	2.871	4.622
Median	5	1	1
SD	10.170	4.477	8.000

Table 6.19: The summary statistics of the indel length distribution of PRANK alignments (PRANK(id)) used in the study is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by ProPIP with k=1 under STFT with fiter: welch and filter size: 128. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

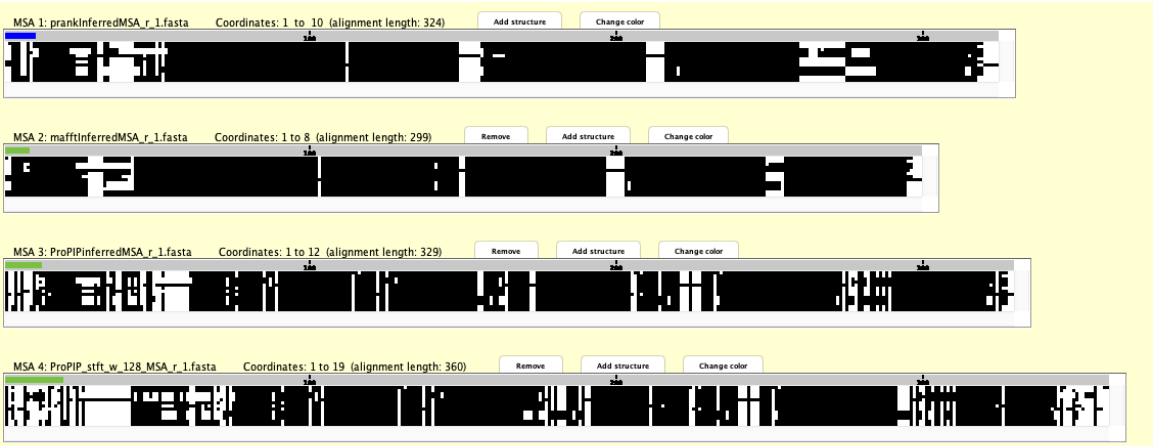


Figure 6.37: The Pixel Plot[1] (Section 5.5). A protein alignment reconstructed by PRANK v.170427 (MSA 1) is compared with MSA's generated by MAFFT v7.453 (MSA 2), ProPIP with k=1 (MSA 3), and ProPIP with k=1 under STFT with fiter: welch and filter size: 128 (MSA 4). Note: black pixel represents Characters and white pixel represents Indels.

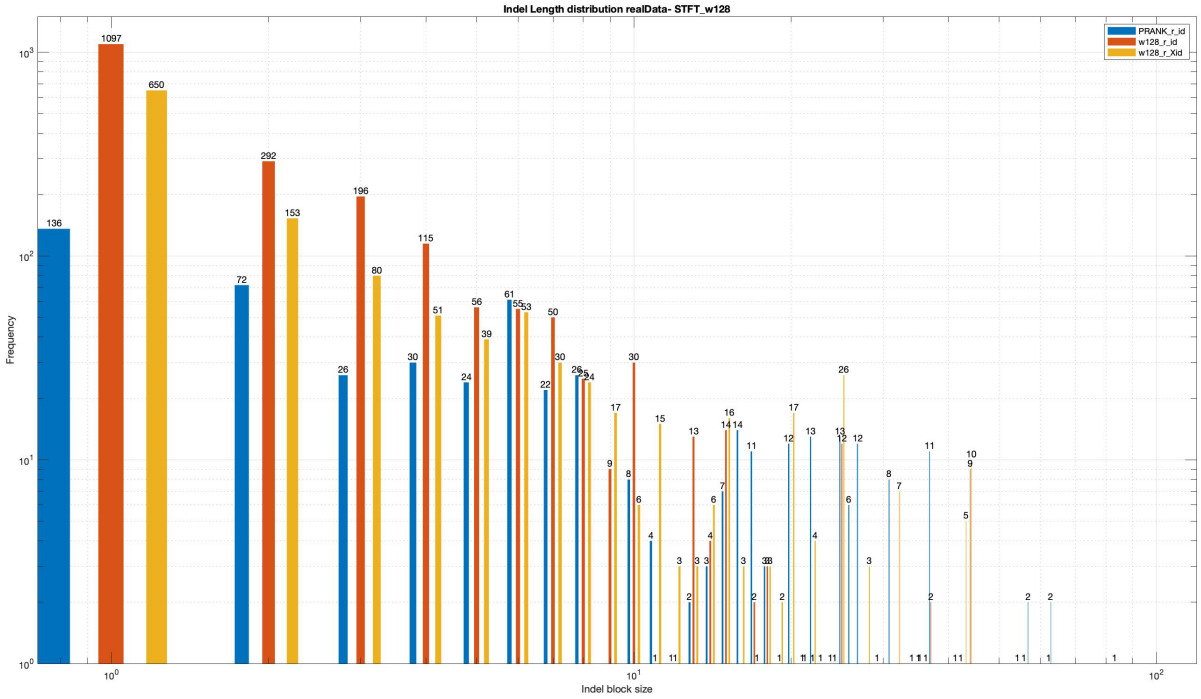


Figure 6.38: The Log-Log Plot. The indel length distribution of PRANK alignments used in the study is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by ProPIP with k=1 under STFT with fiter: welch and filter size: 128. Note: The legend blue represents PRANK indel length distribution, red represents inferred indel length distribution and yellow represents inferred indel block distribution.

6.3.6 Quality check using qscore

A standard quality check on the inferred protein alignments were carried out using the classical quality assessment tool named qscore. qscore evaluates the reference (inferred using PRANK) and observed protein alignments and generates four different scores. We generated box plots showing the results of the scores obtained from the qscore software. This section gives the inference from the box plots (See Figure 6.39 and 6.40) with a short explanation.

Overall, the MAFFT produced the highest Q score values closer to 1. In contrast, our tool ProPIP generated Q score values near to 0.96. On the one hand, it can be observed from the Figures 6.39 and 6.40 that our tool performs better when we use its additional features. For instance, the Q score decreases consistently as k value increases and also the Q score obtained for ProPIP under Discrete Gamma distribution rises (when k is 0.05 and α is 0.10) very slowly as α value increases. Interestingly we obtained a slightly closer Modeler score for STFT to MAFFT, which is an encouraging result for the future work. Furthermore, from Modeler score plot (Figure 6.40) we observed that Discrete Gamma distribution feature in ProPIP is having the higher Modeler score than k-Factor method or the method where we apply both k-Factor and Discrete Gamma distribution in ProPIP, which means that the parameters α and k plays a vital role in generating more closer MSAs to PRANK in this datatype.

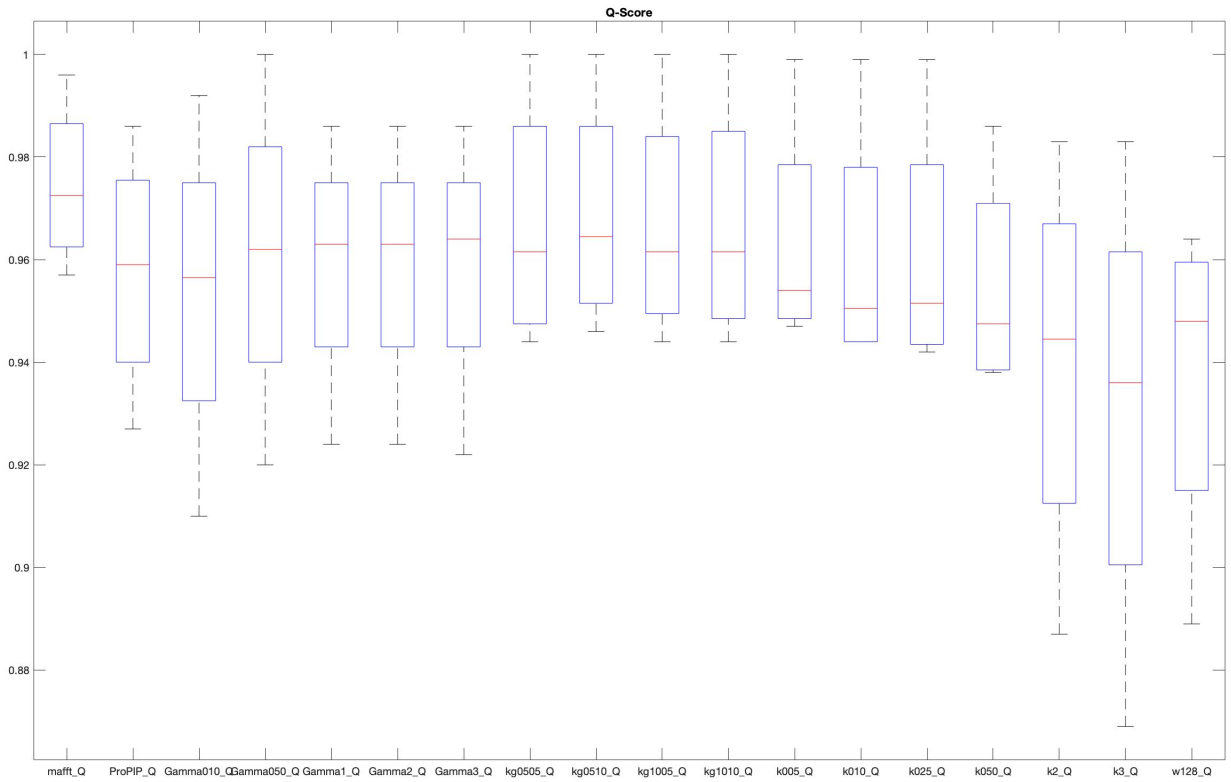


Figure 6.39: Box Plots of Q Scores for real data. Reading from left Q Score box: Box 1: MAFFT, Box 2: ProPIP, Box 3: ProPIP with k=1 under Gamma n=4 and α=0.10, Box 4: ProPIP with k=1 under Gamma n=4 and α=0.50, Box 5: ProPIP with k=1 under Gamma n=4 and α=1, Box 6: ProPIP with k=1 under Gamma n=4 and α=2, Box 7: ProPIP with k=1 under Gamma n=4 and α=3, Box 8: ProPIP with k=0.05 under Gamma n=4 and α=0.05, Box 9: ProPIP with k=0.05 under Gamma n=4 and α=0.10, Box 10: ProPIP with k=0.10 under Gamma n=4 and α=0.05, Box 11: ProPIP with k=0.10 under Gamma n=4 and α=0.10, Box 12: ProPIP with k=0.05, Box 13: ProPIP with k=0.10, Box 14: ProPIP with k=0.25, Box 15: ProPIP with k=0.50, Box 16: ProPIP with k=2, Box 17: ProPIP with k=3, Box 18: ProPIP under STFT with FIR filter and filter size= 129. To note: PRANK alignment is the reference MSA .

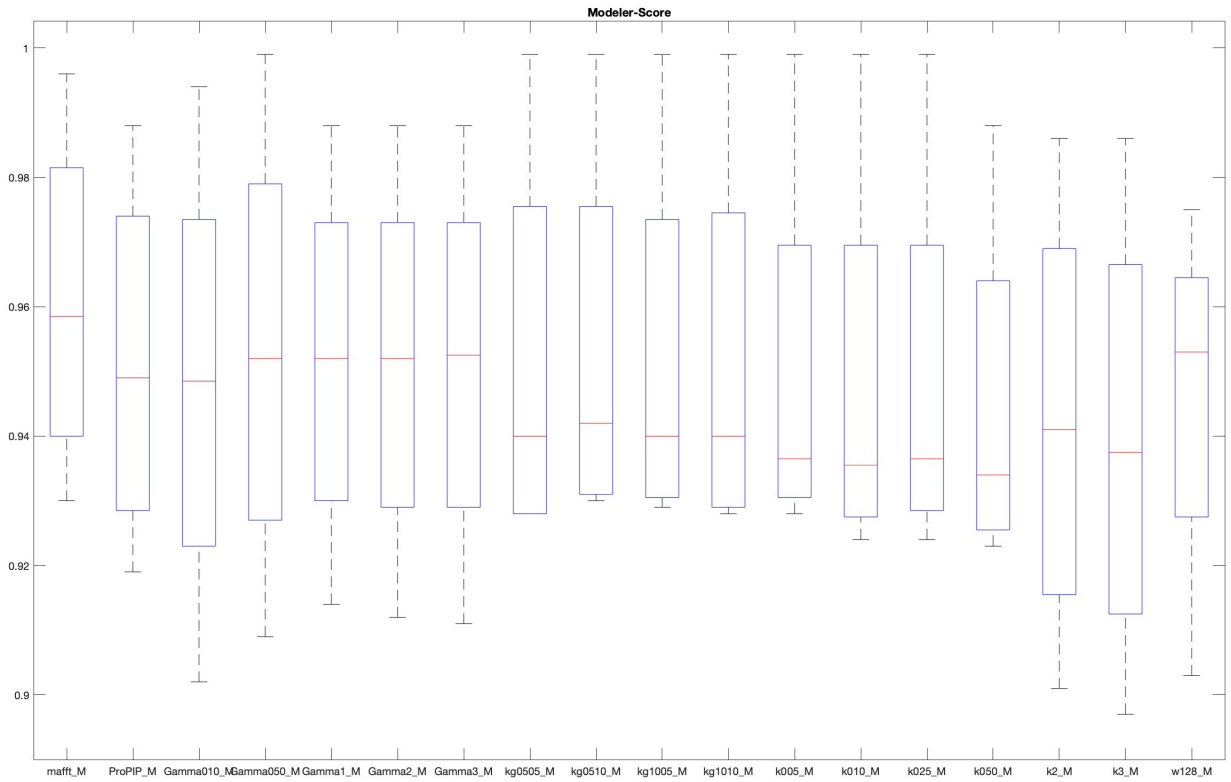


Figure 6.40: Box Plots of TC Scores for real data. Reading from left TC Score box: Box 1: MAFFT, Box 2: ProPIP, Box 3: ProPIP with k=1 under Gamma n=4 and α=0.10, Box 4: ProPIP with k=1 under Gamma n=4 and α=0.50, Box 5: ProPIP with k=1 under Gamma n=4 and α=1, Box 6: ProPIP with k=1 under Gamma n=4 and α=2, Box 7: ProPIP with k=1 under Gamma n=4 and α=3, Box 8: ProPIP with k=0.05 under Gamma n=4 and α=0.05, Box 9: ProPIP with k=0.05 under Gamma n=4 and α=0.10, Box 10: ProPIP with k=0.10 under Gamma n=4 and α=0.05, Box 11: ProPIP with k=0.10 under Gamma n=4 and α=0.10, Box 12: ProPIP with k=0.05, Box 13: ProPIP with k=0.10, Box 14: ProPIP with k=0.25, Box 15: ProPIP with k=0.50, Box 16: ProPIP with k=2, Box 17: ProPIP with k=3, Box 18: ProPIP under STFT with FIR filter and filter size= 129. To note: PRANK alignment is the reference MSA.

Chapter 7

Conclusions and Future works

7.1 Conclusions

In this thesis, we tested the suitability of the single character indel model PIP to infer long indels. To solve this objective, our work proposed a study using simulated data and real data. We used the long indel data simulator INDELible v1.03, four 13-taxa protein alignments and a single character indel data simulator PIPJava provided by the authors of PIP[2]. Our study starts with a comparative study of our aligner ProPIP with two state of art aligners PRANK v.170427 and MAFFT v7.453. Then, to examine the indel information in each data type we study the MSA's using the indel length distribution method and indel block method as explained in Section 5.2 and Section 5.3.

We started by comparing the performance of our aligner ProPIP with MAFFT and PRANK. In PIP data our aligner outperformed the other two aligners with a closer fit in the number of indels and indel length distribution to 'true'. We witnessed the presence of more shorter indels in the alignments inferred from INDELible data and protein sequences (explained in Section 4.3). From the Pixel plot of the inferred MSA's we witnessed that the ProPIP break downs the longer indels into more shorter indels. To further investigate we decided to test all the supplementary features from our aligner ProPIP using the same datasets.

First, we tested the Discrete Gamma distribution technique, which is implemented in our tool to vary the indel rates across the sites. We observed that, by varying the value of parameter α above 1, one cqn infer lower number of indels in the MSA's generated by ProPIP for all data types. However, when the value of α is selected as 3 we received the same statistics as that of default ProPIP, that is ProPIP without additional features. We obtained the same trend when we tested this feature using α values below 1. Moreover, the Q scores of this experiments fall behind the Q score statistic of default ProPIP. Thus, we concluded that this feature is not creating satisfactory results for all data types used in this thesis. Second, a new term which scales the parameter (λ and μ) values of ProPIP was introduced. The parameter k in all data types (simulated and real data) showed that the number of indels in the inferred MSA's can be reduced (also the indel placements) by setting a relatively smaller k value (this study used the lowest k value of 0.05). To note, even though ProPIP under k-Factor attained the same results in PIP data, it attained a closer number of 'true' indels only when k= 2. Except for real data, ProPIP under k-Factor technique achieved a higher Q score value than default ProPIP. Third, we experimented the Discrete Gamma distribution and k-Factor method at the same time in ProPIP. Since we tested this method with smaller value of k and α, our aligner do not acquired adequate results for PIP data. However, interestingly, ProPIP under Discrete Gamma distribution and k-Factor method inferred improved statistics as that of default ProPIP when the parameter values are set to k= 0.05 and α= 0.10. The statistics obtained for real data under this method is supportive. The Q scores obtained from the inferred MSAs using k= 0.05 and α= 0.10 in ProPIP are surpassing the Q score statistics of the default ProPIP.

Fourth, we experimented the Stochastic Backtracking DP algorithm in INDELible and PIP data. The statistics from both datasets suggest that, by setting lower value for distortion parameter T, one can increase the number of indels in the inferred MSAs and at the same time we observed the decrease in the maximum indel length. The Q score withhold this observation by giving a high Q score value when the value of T is lower (in this study we tested a lowest T value of 1). However, the SBDP algorithm is not preferred over the default ProPIP due to the presence of huge indels and unsuitable statistics in its output.

Finally another feature in the ProPIP called Short Time Fourier Transform is examined. The *welch* window function with window size 128 produced competing results to default ProPIP. The results under PIP data is interesting, because the inferred indels and 'true' indels are very close to each other, differ by a small value. But, the presence of extremely longer indels (maximum of 111) makes them distinct. A similar statistics is also achieved from INDELible data and real data with extreme longer indels. Thus, more tests are required using other window functions and window sizes.

To conclude, the overall thesis work show that by using parameters k and α, our aligner (which is based on a single character indel model PIP) can be tuned to infer longer indels. This is an encouraging and motivating milestone of our aligner ProPIP. The future works based on this conclusions are explained in the following section.

7.2 Future works

To understand the dynamics of indels, we have conducted many different experiments on our aligner ProPIP along with a comparative study using two traditional aligners MAFFT and PRANK. Many different parameters and experiments have been left for the future due to time constraints. Future work concerns deeper analysis of particular features in INDELible v1.03, different supplementary features implemented in our tool ProPIP, new proposals to experiment the features in ProPIP and more tests in real data.

There are several boundaries that we set to maintain the structure and focus of the thesis. In detail, when we designed the MSA evaluation methods (Chapter 5) I selected the values for the parameters k, α, T and window function (also window size) randomly. For the future study using this tools it would be interesting to consider more parameter values with different importance. For instance, we witnessed that k is improving the alignments reconstructed using our aligner ProPIP, therefore adding more values or designing an algorithm to find the optimal k is recommended. The results of the Discrete Gamma distribution experiment do not seem to be satisfactory, and further study is still required in order to understand the behavior of the parameter α in our aligner. Moreover, in the case of SBDP algorithm an experiment using low distortion value T is needed to make conclusions about this feature. And, due to computational complexity occurred during the simulation of this algorithm we avoided the test using real data. To add up, the STFT feature from ProPIP includes certain other window functions and window sizes. Therefore, instead of experimenting with only one window function and window size one could conduct a study using other functions, which are already implemented in it. Furthermore, the use of more specific values for k and α at the same time in ProPIP could be investigated since they have an important influence on the results obtained from 13 taxa protein sequences (real data explained in Chapter 4).

Concerning the methods to understand the dynamics of indels, more sophisticated methods to understand how the indel break down is occurring in ProPIP can also improve the results of future research. There are other parameter which could be exploited to see the indel placements such as using a wider range of η (asymptotic expected sequence lengths), replicates and ζ (indel intensities) could shed more light on the possible reasons for the results of this thesis work and could suggest further improvements for the process and algorithm.

Bibliography

[1] C. L. Anderson, C. L. Strope, and E. N. Moriyama. SuiteMSA: Visual tools for multiple sequence alignment comparison and molecular sequence simulation. *BMC Bioinformatics*, 12(1):184, 2011.

[2] A. Bouchard-Cŏté and M. I. Jordan. Evolutionary inference via the Poisson Indel Process. *Proceedings of the National Academy of Sciences of the United States of America*, 110(4):1160–1166, 2013.

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

[4] M. J. F. Dower WJ Ragsdale CW. Nucleic Acids Research erythocytes Nucleic Acids Research. *Nucleic Acids Research*, 9(20):2589–2598, 1981.

[5] V. D. Emmanuelle Lerat and N. A. Moran. Protein alignment information- https://doi.org/10.1371/journal.pbio.0000019.st001.

[6] W. Fletcher and Z. Yang. INDELible: A flexible simulator of biological sequence evolution. *Molecular Biology and Evolution*, 26(8):1879–1888, 2009.

[7] M. Gil and M. Anisimova. Methodologies for Phylogenetic Inference. *eLS*, pages 1–5, 2015.

[8] G. H. Gonnet, M. A. Cohen, and S. A. Benner. Exhaustive matching of the entire protein sequence database. *Science*, 256(5062):1443–1445, 1992.

[9] K. Katoh. MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform. *Nucleic Acids Research*, 30(14):3059–3066, 2002.

[10] K. Katoh and D. M. Standley. MAFFT multiple sequence alignment software version 7: Improvements in performance and usability. *Molecular Biology and Evolution*, 30(4):772–780, 2013.

[11] I. Kosmidis. Bias in parametric estimation: Reduction and useful side-effects. *Wiley Interdisciplinary Reviews: Computational Statistics*, 6(3):185–196, 2014.

[12] L. Gatti & M. Maiolo. ProPIP, castor_aligner, manual, https://github.com/acg-team/castor_aligner, 2017.

[13] E. Lerat, V. Daubin, and N. A. Moran. From gene trees to organismal phylogeny in prokaryotes: The case of the γ-Proteobacteria. *PLoS Biology*, 1(1):101–109, 2003.

[14] A. Löytynoja and N. Goldman. An algorithm for progressive multiple alignment of sequences with insertions. *Proceedings of the National Academy of Sciences of the United States of America*, 102(30):10557–10562, 2005.

[15] M. Maiolo, X. Zhang, M. Gil, and M. Anisimova. Progressive multiple sequence alignment with indel evolution. *BMC Bioinformatics*, 19(1):1–8, 2018.

[16] D. b. M. Massimo. https://serval.unil.ch/en/notice/serval:BIB_D24577D3A885, 2019.

[17] U. Mückstein, I. L. Hofacker, and P. F. Stadler. Stochastic pairwise alignments. *Bioinformatics*, 18:S153–S160, 2002.

[18] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, 1970.

[19] Qscore multiple sequence alignment scoring software. https://www.drive5.com/qscore/.

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

[21] M. L. Tress, D. Jones, and A. Valencia. Predicting reliable regions in protein alignments from sequence profiles. *Journal of molecular biology*, 330(4):705–718, 2003.

[22] L. Wang and T. Jiang. On the Complexity of Multiple Sequence Alignment. *Journal of Computational Biology*, 1(4):337–348, 1994.

[23] Z. Yang. Among-site rate variation and its impact on phylogenetic analyses. *Trends in Ecology and Evolution*, 11(9):367–372, 1996.

APPENDICES

Appendix A

Simulating data in INDELible

```
#Step1 is to manupulate the Guide tree.
#In order to use this as the input tree for INDELible

#Now at the PIPJava Trees
#*********************************************************************
setwd("/Users/pouloeld/Documents/INDELIble/INDELiIble_SIM/
                                    INDELible_sim_NewickTrees")
library(phytools)
library(gtools) #Use this to use mixed sort to order strings
                #in order 0 1 2 3
                #instead of [0 1 10 11 20 21]
library(readtext)

treepath="/Users/pouloeld/Documents/INDELIble/INDELiIble_SIM/INDELible
           _____sim_NewickTrees/Trees"
treenames = list.files(treepath, pattern="*.newick")
treenames= mixedsort(treenames,decreasing = TRUE)

c= 0
for(i in treenames){
  GuideTree= read.tree(file=i)
  write.tree(GuideTree,file= paste0("sim-",c,sep= ".tree"))
  #New empty control files are created here
  #sapply(paste0("control-",c, ".txt"), file.create)
  c= c+1
}
sapply(paste0("control",".txt"), file.create)
#*********************************************************************
#Moving to Control file now
controlnames = list.files(pattern="control.txt")
#controlnames= mixedsort(controlnames,decreasing = TRUE)
shorttrees = list.files(pattern="*.tree")
shorttrees= mixedsort(shorttrees,decreasing = TRUE)

#Content (fixed)
#Content (dynamic) Tree block with the data from above
#l= length(controlnames)

#append 1:100
#naming 0:99

sink(controlnames[1])

cat("[TYPE] AMINOACID 1")
cat("\n")
cat("\n")

cat("[SETTINGS]")
cat("\n")
cat("   [ancestralprint]           NEW")
cat("\n")
cat("   [output]                   FASTA")
cat("\n")
```

```
cat("   [phylipextension]           phy")
cat("\n")
cat("   [nexusextension]           new")
cat("\n")
cat("   [fastaextension]           fasta")
cat("\n")
cat("   [randomseed]               1234567")
cat("\n")
cat("   [printrates]               TRUE")
cat("\n")
cat("   [insertaslowercase]        FALSE")
cat("\n")
cat("   [markdeletedinsertions]    TRUE")
cat("\n")
cat("   [printcodonsasaminoacids]  FALSE")
cat("\n")
cat("   [fileperrep]               TRUE")
cat("\n")
cat("\n")

cat("[MODEL] WAGModel")
cat("\n")
cat("   [submodel]                 WAG")
cat("\n")
cat("   [indelmodel]               POW 1.7")
cat("\n")
cat("   [indelrate]                0.05")
cat("\n")
cat("\n")

#file.show(controlnames[1])
c=0
for(j in 1:100){
  #here we need to add tree from shorttrees[j] which is a tree file
  treeastext= readtext(shorttrees[j],verbosity = 0)
  trr= treeastext$text
  trr= unlist(strsplit(trr, split='root', fixed=TRUE))
  trr= paste0(trr[1],trr[2])
  trr0= unlist(strsplit(trr, split='internal_0', fixed=TRUE))
  trr0= paste0(trr0[1],trr0[2])
  trr1= unlist(strsplit(trr0, split='internal_1', fixed=TRUE))
  trr1= paste0(trr1[1],trr1[2])
  trr2= unlist(strsplit(trr1, split='internal_2', fixed=TRUE))
  trr2= paste0(trr2[1],trr2[2])
  trr3= unlist(strsplit(trr2, split='internal_3', fixed=TRUE))
  trr3= paste0(trr3[1],trr3[2])
  trr4= unlist(strsplit(trr3, split='internal_4', fixed=TRUE))
  trr4= paste0(trr4[1],trr4[2])
  trr5= unlist(strsplit(trr4, split='internal_5', fixed=TRUE))
  trr5= paste0(trr5[1],trr5[2])

  write(trr5, file=paste0("sim_",c,".newick"))
  cat(paste0(("[TREE]   t_"),j-1),trr5)
  cat("\n")
  cat("\n")
  c= c+1
}
```

```
for(j in 1:100){
cat(paste0(("[PARTITIONS] ________Pname_"),j-1),paste0(("[t_"),j-1),
                                                    paste0("WAGModel_200]"))
cat("\n")
cat("\n")
}

cat("[EVOLVE]")
cat("\n")
for(j in 1:100){
cat(    paste0(("Pname_"),j-1),paste("1"),paste0(("out_"),j-1))
cat("\n")
}
sink()
```

Appendix B

Generating guide trees for real data using PRANK

```
#This generates tree for the given alignment using PRANK aligner.

for nF in {1..4};
do
        prank -d=./realData_$nF.fasta -o=./real_$nF.newick -treeonly
done
```

MAFFT tree generation for INDELible, PIP, real data

```
#This bash script generates mafft tree for the given newick tree.
#Example newick tree:
#((1:0.1,2:0.1):0.3,(3:0.3,(4:0.2,5:0.2):0.1):0.1)
#Name of each leaf must be an Integer
#Branch length must be provided.
#Tree has to be rooted tree
# To run: use the ruby ibterpreter and  newick2mafft.rb script

# INDELible input newick tree
for nF in {1..100};
do
        ruby ./newick2mafft.rb 1.0 ./tree_I_$nF.newick > ./tree_I_$nF.mafft
done

# PIPJava input newick tree
for treefile in {1..100};
do
        ruby ./newick2mafft.rb 1.0 ./tree_P_$nF.newick > ./tree_I_$nF.mafft
done

# real data input newick tree
for treefile in {1..4};
do
        ruby ./newick2mafft.rb 1.0 ./tree_r_$nF.newick > ./tree_r_$nF.mafft
done
```

Appendix C

Generating parameter values given Tree and MSA

```
%% THIS FILE IS USED TO GENERATE parameters (lambda&mu)
% When TRUE MSA and TREE is provided.
clc
clear all

%% params used in Simulation model
nTaxa= 8;
nIter= 100;
E= 200;


%% Input data Tools
%1 INDELible/I
%2 ProPIP/P
%3 realData/r

%% Add all paths here
% Include all functions of ProPIP
addpath('/Users/pouloeld/Documents/ProPIP/Params_Generation/scripts')
%Include Inputs
addpath('/Users/pouloeld/Documents/ProPIP/Params_Generation/inputs/
____________________INDELible_ins/gappedMSA_INDELible')
addpath('/Users/pouloeld/Documents/ProPIP/Params_Generation/inputs/
____________________INDELible_ins/newicktree_INDELible')
%Include matfiles needed for generating     &
addpath('/Users/pouloeld/Documents/ProPIP/Params_Generation/outputs/
________________________INDELible_outs/mat_functions')


%% Inputs Contains Inputs from different tools such as:
%1 INDELible_ins
%2 ProPIP_ins
%3 realData_ins

% all directory contains same items as mentioned below:
%1 gappedMSA_INDELible
%2 newicktree_INDELible

%% Tool:- INDELible

%1. MSAs
msaPath= '/Users/pouloeld/Documents/ProPIP/Params_Generation/inputs/
____________________INDELible_ins/gappedMSA_INDELible';
fastafiles= dir(fullfile(msaPath,'*.fasta'));
allmsaNames= {fastafiles.name};
mk= regexp(allmsaNames,'((?<=out_)\d*)|((?<=_TRUE1_))','match');
[~,mii]= sortrows(str2double(cat(1,mk{:})));
allmsaNames= allmsaNames(mii);

%2.1 Trees
treePath= '/Users/pouloeld/Documents/ProPIP/Params_Generation/inputs/
____________________INDELible_ins/newicktree_INDELible';
treefiles= dir(fullfile(treePath,'*.newick'));
```

```
alltreeNames= {treefiles.name};
tk= regexp(alltreeNames,'((?<=sim_)\d*)','match');
[~,tii]= sortrows(str2double(cat(1,tk{:})));
alltreeNames= alltreeNames(tii);

%2.2 Tree Head Names used in side the loop

%% To write into another folder in another location
% Read all MSAs and convert msa2matrix then use getMSAfeatures function on
% this matrix to get the details for each MSA. This is then written into a
% file
%% Create Outputs locations Outputs contains:

%1 INDELible_outs
%2 ProPIP_outs
%3 realData_outs
% all directory contains same items as mentioned below:

%1 column_counter
%2 exp_seq_len
%3 lambda_mu
%4 mat_function

column_counter_path= '/Users/pouloeld/Documents/ProPIP/Params_Generation/
    ____________________outputs/INDELible_outs/column_counter';
exp_seq_len_path= '/Users/pouloeld/Documents/ProPIP/Params_Generation/
    ____________________outputs/INDELible_outs/exp_seq_len';
lambda_mu_path= '/Users/pouloeld/Documents/ProPIP/Params_Generation/
    ____________________outputs/INDELible_outs/lambda_mu';
mat_functions_path= '/Users/pouloeld/Documents/ProPIP/Params_Generation/
    ____________________outputs/INDELible_outs/mat_functions';

%% Base Names of the Output files
column_counter_file= 'column_counter_I_';
exp_seq_len_file= 'exp_seq_I_';
lambda_mu_file= 'lambda_mu_I_';
mat_functions_file= 'mat_functions_I_';

%% This loop will generate 2 files in different locations on each iteration

for i=1:nIter
    %exp_seq_len
    fid_exp= fopen(strcat(exp_seq_len_path,'/',exp_seq_len_file,
                          num2str(i)),'w');
    disp(strcat('Printing_exp_seq_len_files...',num2str(i)))
    filenameMSA= char(allmsaNames(i));
    msa= fastaread(filenameMSA);
    mat= msa2matrix(msa);
    exp_seq_len_value= getMSAfeatures(mat);
    fprintf(fid_exp,'%d\n',exp_seq_len_value);


    fid_column_counter= fopen(strcat(column_counter_path,'/',
                column_counter_file,num2str(i)),'w');
    disp(strcat('Printing_column_counter_files...',num2str(i)))
    column_counter= zeros(1,2^nTaxa-1);

    for j=1:size(mat,2)
```

```
        colm= mat(:,j);
        n= col2num(colm, filenameMSA, j);
        if n>0
            column_counter(n)= column_counter(n)+1;
        end
    end

    for j= 1:length(column_counter)
        if column_counter(j)>0
            fprintf(fid_column_counter,'%d , %d \n',j ,
                                column_counter(j));
        end
    end
    fclose(fid_column_counter);
end
 fclose(fid_exp);

disp('End of Section I')

%% SECTION II Generate matlab functions files


for k=1:nIter
    %% Expected length generated on SECTION I
    exp_seq_filesName= strcat(exp_seq_len_path,'/',
                        exp_seq_len_file,num2str(k));
    esl= load(exp_seq_filesName);

    %% Column counter generated on SECTION I
    columns_counter_filesName= strcat(column_counter_path,'/',
                          column_counter_file,num2str(k));
    columns_count= load(columns_counter_filesName);
    ncomb= size(columns_count,1);
    msa2= char(zeros(nTaxa,ncomb));

    %% Trees from Inputs
    filenameTREE= char(alltreeNames(k));

    % nwk to tree
    D= nwk2tree(filenameTREE);
    nodes= cell(length(D),1);
    count= 1;
    for m= 1:length(D)
      nodes{m}= D(m);
      if contains(nodes{m}.name,'internal')
          nodes{m}.name= strcat('Branch',num2str(count));
          count= count+1;
      end
      if isempty(nodes{m}.Parent)
          root= nodes{m};
          nodes{m}.name= 'root';
      end
    end

    for m= 1:length(D)
        if strcmp(D(m).name,'Branch 1')
            D(m).name= 'Branch1';
        elseif strcmp(D(m).name,'Branch 2')
```

```
            D(m).name= 'Branch2';
        elseif strcmp(D(m).name,'Branch 3')
            D(m).name= 'Branch3';
        elseif strcmp(D(m).name,'Branch 4')
            D(m).name= 'Branch4';
        elseif strcmp(D(m).name,'Branch 5')
            D(m).name= 'Branch5';
        elseif strcmp(D(m).name,'Branch 6')
            D(m).name= 'Branch6';
        end
    end



%% MSA Headers from Inputs
filenameMSA= char(allmsaNames(k));
msa= fastaread(filenameMSA);
headnameMSA= convertCharsToStrings({msa.Header});

%% create m files
fid_mat= fopen(strcat(mat_functions_path,'/',mat_functions_file,
                num2str(k),'.m'),'w');

%% Scripts into m file
disp(strcat('Printing m files...',num2str(k)))

%1 Function Name and Arguments
funcname= strcat('[lambda,mu]=',mat_functions_file,num2str(k),'()');
fprintf(fid_mat,'function %s\n', funcname);
fprintf(fid_mat,'\n');

%2 Tree details
for n=1:length(nodes)
    fprintf(fid_mat,'%s.bl=%f;\n',nodes{n}.name,nodes{n}.br_l);
end
fprintf(fid_mat,'%%%%\n');

for p=1:ncomb
    s= dec2bin(p,2);
    s= pad(s,nTaxa,'left','0');
    for q= 1:nTaxa
        if s(q)=='0'
            ch='-';
        else
            ch='A';
        end
        msa2(q,p)=ch;
    end
end

%3
fprintf(fid_mat,'fun = @(x)(...\n');
fprintf(fid_mat,'[');
fprintf(fid_mat,'x(1)/x(2)-%d;\n', esl);

for p=1:ncomb
    msa2_col= msa2(:,p);
    for n= 1:length(nodes)
```

```
            for r= 1:length(msa2_col)
                if nodes{n}.name == headnameMSA(r)
                    nodes{n}.ch = msa2_col(r);
                    break
                end
            end
        end

        numchar= sum(msa2_col~= '-'); %tilde
        assign_setA(root,numchar);

    %Variables needed for poi_distr_column function (see below)
    lambda= 0.1;
    mu= 0.1;
    tau=compute_tau(root);
    setIota_rec(root,tau,mu,true)
    setBeta_rec(root,mu,true)

        recflag(root) % RECURSION


        poi_distr_column(root,root,lambda,mu,nodes)

        txtS= root.txt;
        str= txtS{1};
        str= strrep(str,'lambda','x(1)');
        str= strrep(str,'mu','x(2)');
        fprintf(fid_mat,'%s',str);

        for t=2:size(txtS,1)
            str= txtS{t};
            str= strrep(str,'lambda','x(1)');
            str= strrep(str,'mu','x(2)');

            fprintf(fid_mat,'+...\n');
            fprintf(fid_mat,'%s',str);
        end

        fprintf(fid_mat,'-%f;\n',columns_count(p,2));

        cleanTree(root)

    end

    fprintf(fid_mat,']...\n');
    fprintf(fid_mat,');\n\n');
    fprintf(fid_mat,'%%%%\n');

    %4
    fprintf(fid_mat, 'E=%f;\n',E);
    fprintf(fid_mat, 'x0=[E/10,1/10];\n');

    %5
    fprintf(fid_mat,'options.Algorithm=␣\''levenberg-marquardt\'';\n');
    fprintf(fid_mat,'options.MaxFunEvals=␣1000;\n');
    fprintf(fid_mat,'options.Display=␣\''off\'';\n');
    fprintf(fid_mat,'s=␣lsqnonlin(fun,x0,[],[],options);\n');
    fprintf(fid_mat,'lambda=␣s(1);\n');
```

```
    fprintf(fid_mat,'mu=%s(2);\n');

    fclose(fid_mat);

end

disp('End of Section II')

%% SECTION III
% Generating      &

for u=1:nIter
    mat_functions_filesName= strcat(mat_functions_file,num2str(u));

    %% Create lambda_mu files
    fid_lambda_mu= fopen(strcat(lambda_mu_path,'/',lambda_mu_file,
            num2str(u),'.txt'),'w');
    disp(strcat('Printing    &   values...',num2str(u)))
    [lambda1,mu1]= feval(mat_functions_filesName); % Calling m functions
    fprintf(fid_lambda_mu,'%f,%f\n',lambda1,mu1);
    fclose(fid_lambda_mu);

end

disp('End of    &   Generation')
```

Appendix D

Indel length analysis in INDELible data

```
clc
clear all

% Date: 15.04.2020
%% This process is with INDELible data with cut-off, m=20
%% Add paths
addpath("/Users/pouloeld/Documents/Statistics2/INDELible/True/m_20")
addpath("/Users/pouloeld/Documents/Statistics2/INDELible/MAFFT/m_20")
addpath("/Users/pouloeld/Documents/Statistics2/INDELible/PRANK/m_20")
addpath("/Users/pouloeld/Documents/Statistics2/INDELible/ProPIP/m_20")


%% Readin TRUE MSA
%INDELible
True_I_Path="/Users/pouloeld/Documents/Statistics2/INDELible/True/m_20";
True_I_files= dir(fullfile(True_I_Path,'*_TRUE_1.fasta'));
True_I_Names= {True_I_files.name};
tk_i= regexp(True_I_Names,'((?<=out_)\d*)|((?<=_TRUE_1))','match');
[~,ti]= sortrows(str2double(cat(1,tk_i{:})));
True_I_Names= True_I_Names(ti);
nIter_I= length(True_I_Names);

%% Readin all Inferred MSAs
% MAFFT
Inf_I_Path1= "/Users/pouloeld/Documents/Statistics2/INDELible/MAFFT/m_20";
Inf_I_files1= dir(fullfile(Inf_I_Path1,'*.fasta'));
Inf_I_Names1= {Inf_I_files1.name};
tk_i1= regexp(Inf_I_Names1,'((?<=mafftInferredMSA_I_)\d*)','match');
[~,ti1]= sortrows(str2double(cat(1,tk_i1{:})));
Inf_I_Names1= Inf_I_Names1(ti1);
nIter_I1= length(Inf_I_Names1);
% PRANK
Inf_I_Path2= "/Users/pouloeld/Documents/Statistics2/INDELible/PRANK/m_20";
Inf_I_files2= dir(fullfile(Inf_I_Path2,'*.fasta'));
Inf_I_Names2= {Inf_I_files2.name};
tk_i2= regexp(Inf_I_Names2,'((?<=prankInferredMSA_I_)\d*)','match');
[~,ti2]= sortrows(str2double(cat(1,tk_i2{:})));
Inf_I_Names2= Inf_I_Names2(ti2);
nIter_I2= length(Inf_I_Names2);
% ProPIP
Inf_I_Path3= "/Users/pouloeld/Documents/Statistics2/INDELible/ProPIP/m_20";
Inf_I_files3= dir(fullfile(Inf_I_Path3,'*.fasta'));
Inf_I_Names3= {Inf_I_files3.name};
tk_i3= regexp(Inf_I_Names3,'((?<=ProPIPinferredMSA_I_)\d*)','match');
[~,ti3]= sortrows(str2double(cat(1,tk_i3{:})));
Inf_I_Names3= Inf_I_Names3(ti3);
nIter_I3= length(Inf_I_Names3);


%% INDEL length per True_MSA

%% INDELible
count=0;
```

```
True_I_count= [];
for namesI=1:nIter_I
    True_I_MSA= fastaread(True_I_Names(namesI));
    True_I_countMSA= [];
    for nTaxa=1:8
        for seqL=1:length(True_I_MSA(nTaxa).Sequence)
            if True_I_MSA(nTaxa).Sequence(seqL) == '-'
                count= count+1;
            elseif True_I_MSA(nTaxa).Sequence(seqL) ~= '-'
                if count ~= 0
                    True_I_count= [True_I_count,count];
                    True_I_countMSA= [True_I_countMSA,count];
                    count=0;
                end
            end
        end
        if count ~=0
            True_I_count= [True_I_count,count];
            True_I_countMSA= [True_I_countMSA,count];
            count=0;
        end
    end
end

    figure,
    histogram(True_I_countMSA,'Normalization','probability')
    title(strcat('True-INDELible_cutoff20_MSA',int2str(namesI)))
    xlabel('Indel_length')


%% INDEL length per Inf_MSA

%% MAFFT
count=0;
Inf_I_count1= [];
for namesI1=1:nIter_I1
    Inf_I_MSA1= fastaread(Inf_I_Names1(namesI1));
    Inf_I_countMSA1= [];
    for nTaxa=1:8
        for seqL=1:length(Inf_I_MSA1(nTaxa).Sequence)
            if Inf_I_MSA1(nTaxa).Sequence(seqL) == '-'
                count= count+1;
            elseif Inf_I_MSA1(nTaxa).Sequence(seqL) ~= '-'
                if count ~= 0
                    Inf_I_count1= [Inf_I_count1,count];
                    Inf_I_countMSA1= [Inf_I_countMSA1,count];
                    count=0;
                end
            end
        end
        if count ~=0
            Inf_I_count1= [Inf_I_count1,count];
            Inf_I_countMSA1= [Inf_I_countMSA1,count];
            count=0;
        end
    end

end
```

```
    figure,
    histogram(Inf_I_countMSA1,'Normalization','probability')
    title(strcat('MAFFTInferred_MSA',int2str(namesI1)))
    xlabel('Indel_length')

%% PRANK
count=0;
Inf_I_count2=[];
for namesI2=1:nIter_I2
    Inf_I_MSA2=fastaread(Inf_I_Names2(namesI2));
    Inf_I_countMSA2=[];
    for nTaxa=1:8
        for seqL=1:length(Inf_I_MSA2(nTaxa).Sequence)
            if Inf_I_MSA2(nTaxa).Sequence(seqL) == '-'
                count=count+1;
            elseif Inf_I_MSA2(nTaxa).Sequence(seqL) ~= '-'
                if count ~= 0
                    Inf_I_count2=[Inf_I_count2,count];
                    Inf_I_countMSA2=[Inf_I_countMSA2,count];
                    count=0;
                end
            end
        end
        if count ~=0
            Inf_I_count2=[Inf_I_count2,count];
            Inf_I_countMSA2=[Inf_I_countMSA2,count];
            count=0;
        end
    end

end
    figure,
    histogram(Inf_I_countMSA2,'Normalization','probability')
    title(strcat('PrankInferred_MSA',int2str(namesI2)))
    xlabel('Indel_length')

%% ProPIP
count=0;
Inf_I_count3=[];
maxMSA=0;
for namesI3=1:nIter_I3
    Inf_I_MSA3=fastaread(Inf_I_Names3(namesI3));
    Inf_I_countMSA3=[];
    for nTaxa=1:8
        for seqL=1:length(Inf_I_MSA3(nTaxa).Sequence)
            if Inf_I_MSA3(nTaxa).Sequence(seqL) == '-'
                count=count+1;
            elseif Inf_I_MSA3(nTaxa).Sequence(seqL) ~= '-'
                if count ~= 0
                    Inf_I_count3=[Inf_I_count3,count];
                    Inf_I_countMSA3=[Inf_I_countMSA3,count];
%                     if count>29
%                         maxMSA=namesI3;
%                     end
                    count=0;
                end
            end
        end
```

```
        if count ~=0
            Inf_I_count3= [Inf_I_count3,count];
            Inf_I_countMSA3= [Inf_I_countMSA3,count];
%             if count>29
%                 maxMSA=namesI3;
%             end
            count=0;
        end
    end


end


    figure,
    histogram(Inf_I_countMSA3,'Normalization','probability')
    title(strcat('ProPIPInferred_MSA',int2str(namesI3)))
    xlabel('Indel_length')

stat{1}= [length(True_I_count),max(True_I_count),mean(True_I_count),
                    median(True_I_count),std(True_I_count)];
stat1{1}= [length(Inf_I_count1),max(Inf_I_count1),mean(Inf_I_count1),
                    median(Inf_I_count1),std(Inf_I_count1)];
stat2{1}= [length(Inf_I_count2),max(Inf_I_count2),mean(Inf_I_count2),
                    median(Inf_I_count2),std(Inf_I_count2)];
stat3{1}= [length(Inf_I_count3),max(Inf_I_count3),mean(Inf_I_count3),
                    median(Inf_I_count3),std(Inf_I_count3)];
```

Appendix E

Indel length analysis in PIP data

```
clc
clear all

% Date: 15.04.2020
%% This process is with PIPJava data without reshuffling
%% Add paths
addpath("/Users/pouloeld/Documents/Statistics2/PIPJava/True")
addpath("/Users/pouloeld/Documents/Statistics2/PIPJava/MAFFT")
addpath("/Users/pouloeld/Documents/Statistics2/PIPJava/PRANK")
addpath("/Users/pouloeld/Documents/Statistics2/PIPJava/ProPIP")


%% Readin TRUE MSA
%INDELible
True_P_Path="/Users/pouloeld/Documents/Statistics2/PIPJava/True";
True_P_files= dir(fullfile(True_P_Path,'*_MSA.fasta'));
True_P_Names= {True_P_files.name};
tk_p= regexp(True_P_Names,'((?<=sim-)\d*)|((?<=_MSA))','match');
[~,tp]= sortrows(str2double(cat(1,tk_p{:})));
True_P_Names= True_P_Names(tp);
nIter_P= length(True_P_Names);

%% Readin all Inferred MSAs
% MAFFT
Inf_P_Path1= "/Users/pouloeld/Documents/Statistics2/PIPJava/MAFFT";
Inf_P_files1= dir(fullfile(Inf_P_Path1,'*.fasta'));
Inf_P_Names1= {Inf_P_files1.name};
tk_p1= regexp(Inf_P_Names1,'((?<=mafftInferredMSA_P_)\d*)','match');
[~,tp1]= sortrows(str2double(cat(1,tk_p1{:})));
Inf_P_Names1= Inf_P_Names1(tp1);
nIter_P1= length(Inf_P_Names1);
% PRANK
Inf_P_Path2= "/Users/pouloeld/Documents/Statistics2/PIPJava/PRANK";
Inf_P_files2= dir(fullfile(Inf_P_Path2,'*.fasta'));
Inf_P_Names2= {Inf_P_files2.name};
tk_p2= regexp(Inf_P_Names2,'((?<=prankInferredMSA_P_)\d*)','match');
[~,tp2]= sortrows(str2double(cat(1,tk_p2{:})));
Inf_P_Names2= Inf_P_Names2(tp2);
nIter_P2= length(Inf_P_Names2);
% ProPIP
Inf_P_Path3= "/Users/pouloeld/Documents/Statistics2/PIPJava/ProPIP";
Inf_P_files3= dir(fullfile(Inf_P_Path3,'*.fasta'));
Inf_P_Names3= {Inf_P_files3.name};
tk_p3= regexp(Inf_P_Names3,'((?<=ProPIPinferredMSA_P_)\d*)','match');
[~,tp3]= sortrows(str2double(cat(1,tk_p3{:})));
Inf_P_Names3= Inf_P_Names3(tp3);
nIter_P3= length(Inf_P_Names3);


%% INDEL length per True_MSA

%% INDELible
count=0;
```

```
True_P_count= [];
for namesP=1:nIter_P
    True_P_MSA= fastaread(True_P_Names(namesP));
    for nTaxa=1:8
        for seqL=1:length(True_P_MSA(nTaxa).Sequence)
            if True_P_MSA(nTaxa).Sequence(seqL) == '-'
                count= count+1;
            elseif True_P_MSA(nTaxa).Sequence(seqL) ~= '-'
                if count ~= 0
                    True_P_count= [True_P_count,count];
                    count=0;
                end
            end
        end
        if count ~=0
            True_P_count= [True_P_count,count];
            count=0;
        end
     end
 MsaL_P= seqL;
end


%% INDEL length per Inf_MSA

%% MAFFT
count=0;
Inf_P_count1= [];
for namesP1=1:nIter_P1
    Inf_P_MSA1= fastaread(Inf_P_Names1(namesP1));
    for nTaxa=1:8
        for seqL=1:length(Inf_P_MSA1(nTaxa).Sequence)
            if Inf_P_MSA1(nTaxa).Sequence(seqL) == '-'
                count= count+1;
            elseif Inf_P_MSA1(nTaxa).Sequence(seqL) ~= '-'
                if count ~= 0
                    Inf_P_count1= [Inf_P_count1,count];
                    count=0;
                end
            end
        end
        if count ~=0
            Inf_P_count1= [Inf_P_count1,count];
            count=0;
        end
    end
MsaL_I1= seqL;
end

%% PRANK
count=0;
Inf_P_count2= [];
for namesP2=1:nIter_P2
    Inf_P_MSA2= fastaread(Inf_P_Names2(namesP2));
    for nTaxa=1:8
        for seqL=1:length(Inf_P_MSA2(nTaxa).Sequence)
            if Inf_P_MSA2(nTaxa).Sequence(seqL) == '-'
                count= count+1;
```

```
            elseif Inf_P_MSA2(nTaxa).Sequence(seqL) ~= '-'
                if count ~= 0
                    Inf_P_count2= [Inf_P_count2,count];
                    count=0;
                end
            end
        end
        if count ~=0
            Inf_P_count2= [Inf_P_count2,count];
            count=0;
        end
    end
MsaL_PS2= seqL;
end

%% ProPIP
count=0;
Inf_P_count3= [];
for namesP3=1:nIter_P3
    Inf_P_MSA3= fastaread(Inf_P_Names3(namesP3));
    for nTaxa=1:8
        for seqL=1:length(Inf_P_MSA3(nTaxa).Sequence)
            if Inf_P_MSA3(nTaxa).Sequence(seqL) == '-'
                count= count+1;
            elseif Inf_P_MSA3(nTaxa).Sequence(seqL) ~= '-'
                if count ~= 0
                    Inf_P_count3= [Inf_P_count3,count];
                    count=0;
                end
            end
        end
        if count ~=0
            Inf_P_count3= [Inf_P_count3,count];
            count=0;
        end
    end
MsaL_P3= seqL;
end


stat{1}= [length(True_P_count),max(True_P_count),mean(True_P_count),
                    median(True_P_count),std(True_P_count)];
stat1{1}= [length(Inf_P_count1),max(Inf_P_count1),mean(Inf_P_count1),
                    median(Inf_P_count1),std(Inf_P_count1)];
stat2{1}= [length(Inf_P_count2),max(Inf_P_count2),mean(Inf_P_count2),
                    median(Inf_P_count2),std(Inf_P_count2)];
stat3{1}= [length(Inf_P_count3),max(Inf_P_count3),mean(Inf_P_count3),
                    Pmedian(Inf_P_count3),std(Inf_P_count3)];
```

Appendix F

Indel length analysis in real data

```
clc
clear all

% Date: 15.04.2020
%% This process is with real data
%% Add paths
addpath("/Users/pouloeld/Documents/Statistics2/real/True")
addpath("/Users/pouloeld/Documents/Statistics2/real/MAFFT")
addpath("/Users/pouloeld/Documents/Statistics2/real/PRANK")
addpath("/Users/pouloeld/Documents/Statistics2/real/ProPIP")


%% Readin TRUE MSA
%INDELible
True_r_Path="/Users/pouloeld/Documents/Statistics2/real/True";
True_r_files= dir(fullfile(True_r_Path,'*.fasta'));
True_r_Names= {True_r_files.name};
tk_r= regexp(True_r_Names,'((?<=realData_)\d*)','match');
[~,tr]= sortrows(str2double(cat(1,tk_r{:})));
True_r_Names= True_r_Names(tr);
nIter_r= length(True_r_Names);

%% Readin all Inferred MSAs
% MAFFT
Inf_r_Path1= "/Users/pouloeld/Documents/Statistics2/real/MAFFT";
Inf_r_files1= dir(fullfile(Inf_r_Path1,'*.fasta'));
Inf_r_Names1= {Inf_r_files1.name};
tk_r1= regexp(Inf_r_Names1,'((?<=mafftInferredMSA_r_)\d*)','match');
[~,tr1]= sortrows(str2double(cat(1,tk_r1{:})));
Inf_r_Names1= Inf_r_Names1(tr1);
nIter_r1= length(Inf_r_Names1);
% PRANK
Inf_r_Path2= "/Users/pouloeld/Documents/Statistics2/real/PRANK";
Inf_r_files2= dir(fullfile(Inf_r_Path2,'*.fasta'));
Inf_r_Names2= {Inf_r_files2.name};
tk_r2= regexp(Inf_r_Names2,'((?<=prankInferredMSA_r_)\d*)','match');
[~,tr2]= sortrows(str2double(cat(1,tk_r2{:})));
Inf_r_Names2= Inf_r_Names2(tr2);
nIter_r2= length(Inf_r_Names2);
% ProPIP
Inf_r_Path3= "/Users/pouloeld/Documents/Statistics2/real/ProPIP";
Inf_r_files3= dir(fullfile(Inf_r_Path3,'*.fasta'));
Inf_r_Names3= {Inf_r_files3.name};
tk_r3= regexp(Inf_r_Names3,'((?<=ProPIPinferredMSA_r_)\d*)','match');
[~,tr3]= sortrows(str2double(cat(1,tk_r3{:})));
Inf_r_Names3= Inf_r_Names3(tr3);
nIter_r3= length(Inf_r_Names3);


%% INDEL length per True_MSA

%% INDELible
count=0;
```

```
True_r_count= [];
for namesr=1:nIter_r
    True_r_MSA= fastaread(True_r_Names(namesr));
    for nTaxa=1:13
        for seqL=1:length(True_r_MSA(nTaxa).Sequence)
            if True_r_MSA(nTaxa).Sequence(seqL) == '-'
                count= count+1;
            elseif True_r_MSA(nTaxa).Sequence(seqL) ~= '-'
                if count ~= 0
                    True_r_count= [True_r_count,count];
                    count=0;
                end
            end
        end
        if count ~=0
            True_r_count= [True_r_count,count];
            count=0;
        end
     end
 MsaL_r= seqL;
end


%% INDEL length per Inf_MSA

%% MAFFT
count=0;
Inf_r_count1= [];
for namesr1=1:nIter_r1
    Inf_r_MSA1= fastaread(Inf_r_Names1(namesr1));
    for nTaxa=1:13
        for seqL=1:length(Inf_r_MSA1(nTaxa).Sequence)
            if Inf_r_MSA1(nTaxa).Sequence(seqL) == '-'
                count= count+1;
            elseif Inf_r_MSA1(nTaxa).Sequence(seqL) ~= '-'
                if count ~= 0
                    Inf_r_count1= [Inf_r_count1,count];
                    count=0;
                end
            end
        end
        if count ~=0
            Inf_r_count1= [Inf_r_count1,count];
            count=0;
        end
    end
MsaL_r1= seqL;
end

%% PRANK
count=0;
Inf_r_count2= [];
for namesr2=1:nIter_r2
    Inf_r_MSA2= fastaread(Inf_r_Names2(namesr2));
    for nTaxa=1:13
        for seqL=1:length(Inf_r_MSA2(nTaxa).Sequence)
            if Inf_r_MSA2(nTaxa).Sequence(seqL) == '-'
                count= count+1;
```

```
            elseif Inf_r_MSA2(nTaxa).Sequence(seqL) ~= '-'
                if count ~= 0
                    Inf_r_count2= [Inf_r_count2,count];
                    count=0;
                end
            end
        end
        if count ~=0
            Inf_r_count2= [Inf_r_count2,count];
            count=0;
        end
    end
MsaL_r2= seqL;
end

%% ProPIP
count=0;
Inf_r_count3= [];
for namesr3=1:nIter_r3
    Inf_r_MSA3= fastaread(Inf_r_Names3(namesr3));
    for nTaxa=1:13
        for seqL=1:length(Inf_r_MSA3(nTaxa).Sequence)
            if Inf_r_MSA3(nTaxa).Sequence(seqL) == '-'
                count= count+1;
            elseif Inf_r_MSA3(nTaxa).Sequence(seqL) ~= '-'
                if count ~= 0
                    Inf_r_count3= [Inf_r_count3,count];
                    count=0;
                end
            end
        end
        if count ~=0
            Inf_r_count3= [Inf_r_count3,count];
            count=0;
        end
    end
MsaL_r3= seqL;
end


stat{1}= [length(True_r_count),max(True_r_count),mean(True_r_count),
                    median(True_r_count),std(True_r_count)];
stat1{1}= [length(Inf_r_count1),max(Inf_r_count1),mean(Inf_r_count1),
                    median(Inf_r_count1),std(Inf_r_count1)];
stat2{1}= [length(Inf_r_count2),max(Inf_r_count2),mean(Inf_r_count2),
                    median(Inf_r_count2),std(Inf_r_count2)];
stat3{1}= [length(Inf_r_count3),max(Inf_r_count3),mean(Inf_r_count3),
                    median(Inf_r_count3),std(Inf_r_count3)];
```

Plotting True and Inferred indel length data

```
clc
clear all
%% DATA ANALYSIS- Plotting data from
% INDELible, PIP, real data.
% PRANK, MAFFT, ProPIP, ProPIP + Gamma data
addpath("/Users/pouloeld/Desktop/StatisticsIndeLData")
```

```
mat= dir('*.mat');
for q = 1:length(mat)

    load(mat(q).name);
end

%% INDELible
rng 'default'

edges= [1:1:20];
I1= histcounts(True_I_count,edges);
I2= histcounts(Inf_I_countMAFFT,edges);
I3= histcounts(Inf_I_countPRANK,edges);
I4= histcounts(Inf_I_countProPIP,edges);
I5= histcounts(Inf_I_countgamma0,edges);
I6= histcounts(Inf_I_countgamma1,edges);
I7= histcounts(Inf_I_countgamma2,edges);
I8= histcounts(Inf_I_countgamma3,edges);

figure,
bar(edges(1:end-1),[I1;I2;I3;I4;I5;I6;I7;I8]')
title('INDELible data with cut off=20')
xlabel('Indel Length')
ylabel('Frequency')
legend('True-INDELible-with-cut-off=20','Mafft Inferred',
'Prank Inferred','ProPIP Inferred', 'ProPIP+Gamma(n=4,alpha=0.5)',
'ProPIP+Gamma(n=4,alpha=1.0)','ProPIP+Gamma(n=4,alpha=2.0)',
'ProPIP+Gamma(n=4,alpha=3.0)' )
print('INDELible_IndelLengthStat','-dpdf','-fillpage')

%% PIP data

edges= [1:1:12];
P1= histcounts(True_P_count,edges);
P2= histcounts(Inf_P_countMAFFT,edges);
P3= histcounts(Inf_P_countPRANK,edges);
P4= histcounts(Inf_P_countProPIP,edges);
P5= histcounts(Inf_P_countgamma0,edges);
P6= histcounts(Inf_P_countgamma1,edges);
P7= histcounts(Inf_P_countgamma2,edges);
P8= histcounts(Inf_P_countgamma3,edges);

figure,
bar(edges(1:end-1),[P1;P2;P3;P4;P5;P6;P7;P8]')
title('PIPJava data')
xlabel('Indel Length')
ylabel('Frequency')
legend('True-PIPJava','Mafft Inferred','Prank Inferred','ProPIP Inferred',
'ProPIP+Gamma(n=4,alpha=0.5)','ProPIP+Gamma(n=4,alpha=1.0)',
'ProPIP+Gamma(n=4,alpha=2.0)','ProPIP+Gamma(n=4,alpha=3.0)' )
print('PIP_IndelLengthStat','-dpdf','-fillpage')

%% real data

edges= [1:1:30];
r1= histcounts(True_r_count,edges);
r2= histcounts(Inf_r_countMAFFT,edges);
r3= histcounts(Inf_r_countPRANK,edges);
```

```
r4= histcounts(Inf_r_countProPIP,edges);
r5= histcounts(Inf_r_countgamma0,edges);
r6= histcounts(Inf_r_countgamma1,edges);
r7= histcounts(Inf_r_countgamma2,edges);
r8= histcounts(Inf_r_countgamma3,edges);

figure,
bar(edges(1:end-1),[r1;r2;r3;r4;r5;r6;r7;r8]')
title('real data')
xlabel('Indel Length')
ylabel('Frequency')
legend('real data','Mafft Inferred','Prank Inferred','ProPIP Inferred',
'ProPIP+Gamma(n=4,alpha=0.5)','ProPIP+Gamma(n=4,alpha=1.0)',
'ProPIP+Gamma(n=4,alpha=2.0)','ProPIP+Gamma(n=4,alpha=3.0)'␣)
print('realData_IndelLengthStat','-dpdf','-fillpage')
```

Appendix G

Pixel Plot Analysis- Additional Plots

Figure G.1: The Pixel Plot[1] (Section5.5)). A simulated True MSA using INDELible is compared with MSAs generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), ProPIP with k=1 under Gamma n=4, α=0.10 (MSA 5), ProPIP with k=1 under Gamma n=4, α=0.50 (MSA 6), ProPIP with k=1 under Gamma n=4, α=1 (MSA 7), and ProPIP with k=1 under Gamma n=4, α=2 (MSA 8). Note: Black represents Characters and White represents Indels.

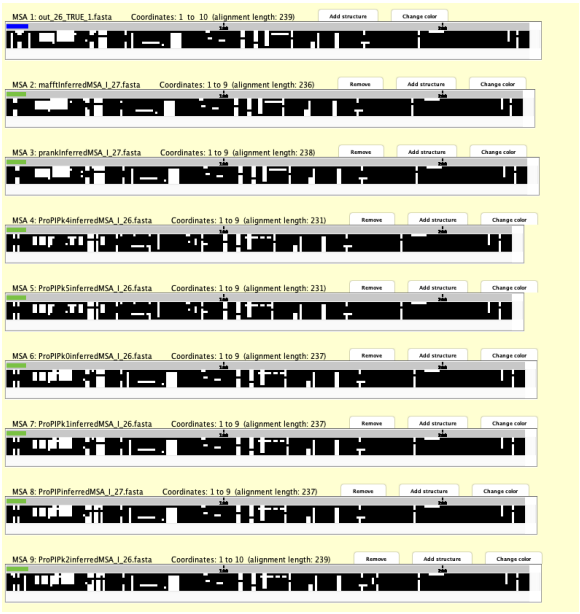


Figure G.2: The Pixel Plot[1] (Section5.5)). A simulated True MSA using INDELible is compared with MSAs generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=0.05 (MSA 4), ProPIP with k=0.10 (MSA 5), ProPIP with k=0.25 (MSA 6), ProPIP with k=0.50 (MSA 7), ProPIP with k=1 (MSA 8), and ProPIP with k=2 (MSA 9). Note: Black represents Characters and White represents Indels.

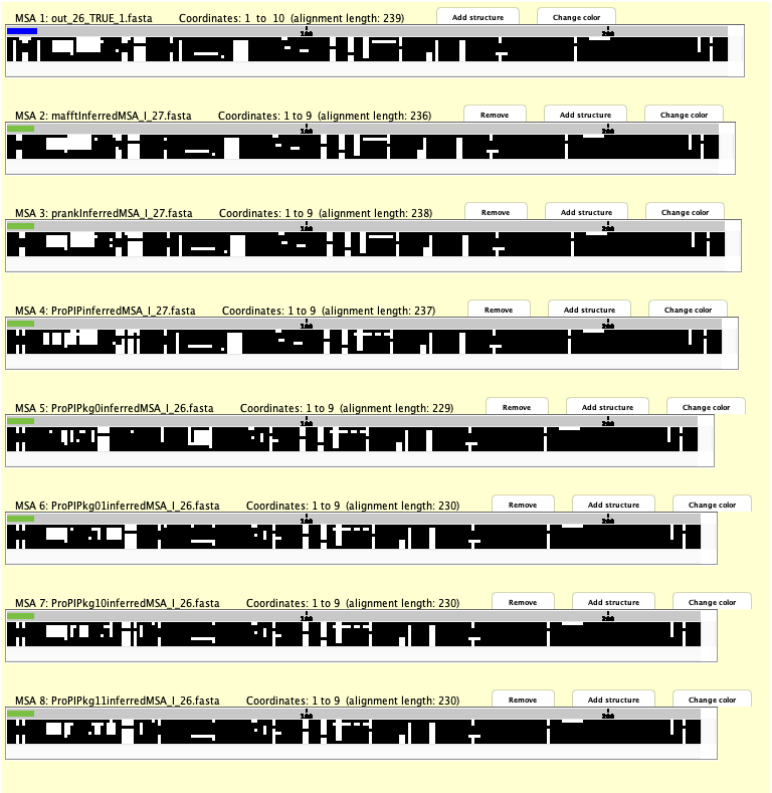


Figure G.3: The Pixel Plot[1] (Section5.5)). A simulated True MSA using INDELible is compared with MSAs generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), ProPIP with k=0.05 under Gamma n=4, α=0.05 (MSA 5), ProPIP with k=0.05 under Gamma n=4, α=0.10 (MSA 6), ProPIP with k=0.10 under Gamma n=4, α=0.05 (MSA 7), and ProPIP with k=0.10 under Gamma n=4, α=0.10 (MSA 8), Note: Black represents Characters and White represents Indels.

Figure G.4: The Pixel Plot[1] (Section5.5)). A simulated True MSA using INDELible is compared with MSAs generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), ProPIP with k=1 under SBDP with T= 1 (MSA 5), ProPIP with k=1 under SBDP with T= 3 (MSA 6), ProPIP with k=1 under SBDP with T= 5 (MSA 7), and ProPIP with k=1 under SBDP with T= 10 (MSA 8). Note: Black represents Characters and White represents Indels.

Figure G.5: The Pixel Plot[1] (Section5.5)). A simulated True MSA using INDELible is compared with MSAs generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), and ProPIP with k=1 under STFT with fiter: welch and filter size: 128 (MSA 5). Note: Black represents Characters and White represents Indels.

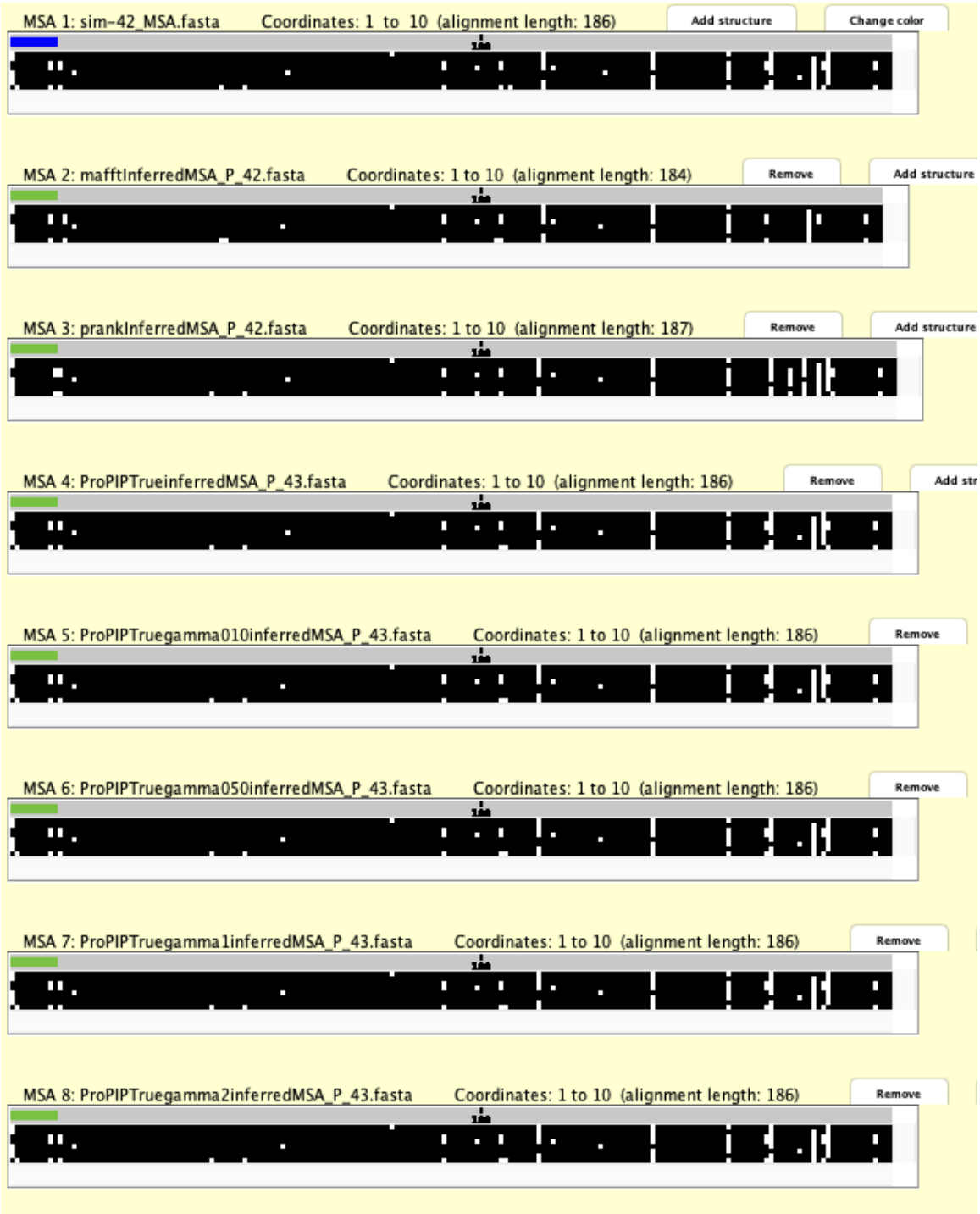


Figure G.6: The Pixel Plot[1] (Section5.5)). A simulated True MSA using INDELible is compared with MSAs generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), ProPIP with k=1 under Gamma n=4, α=0.10 (MSA 5), ProPIP with k=1 under Gamma n=4, α=0.50 (MSA 6), ProPIP with k=1 under Gamma n=4, α=1 (MSA 7), and ProPIP with k=1 under Gamma n=4, α=2 (MSA 8). Note: Black represents Characters and White represents Indels.

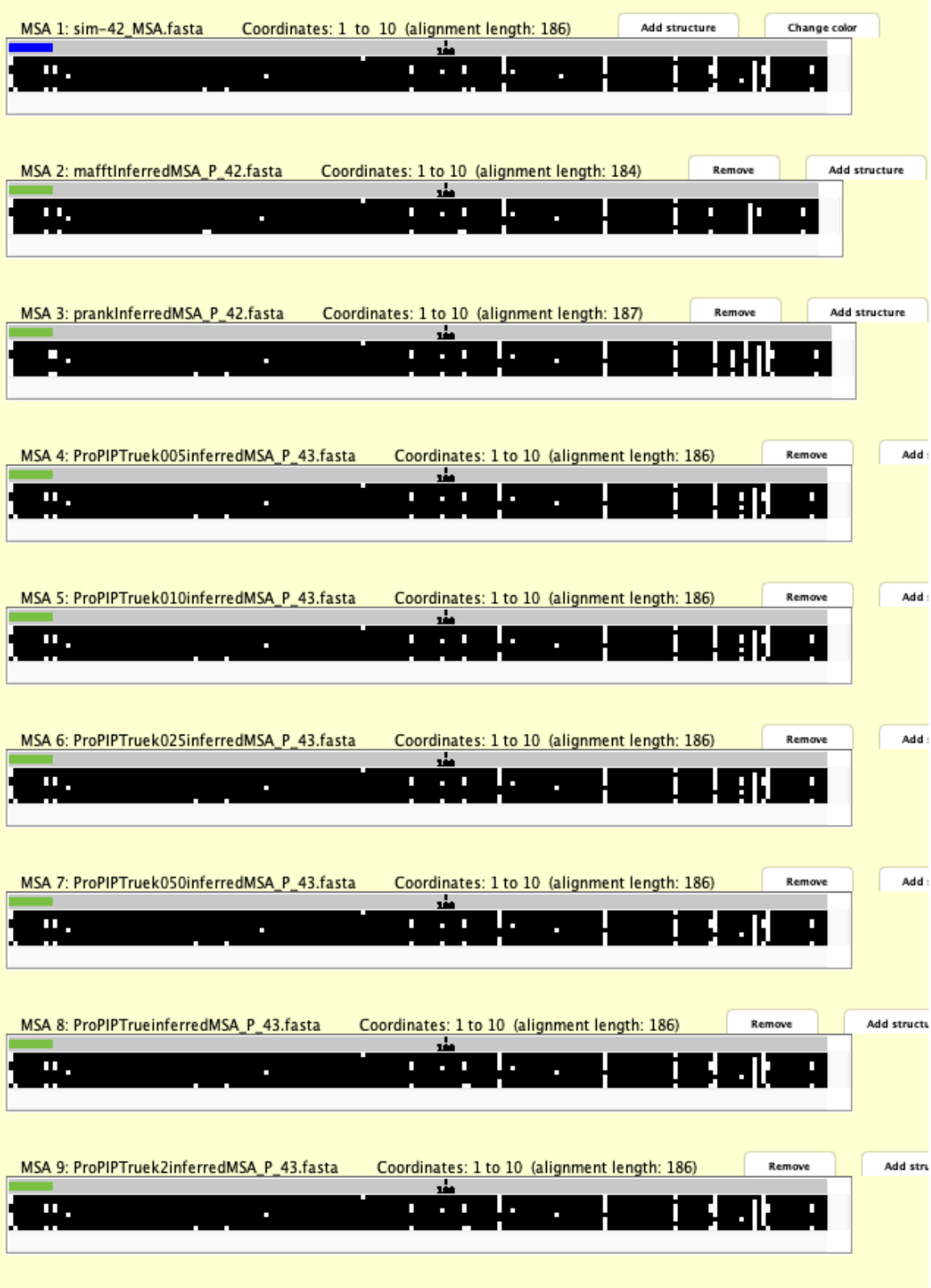


Figure G.7: The Pixel Plot[1] (Section5.5)). A simulated True MSA using PIPJava is compared with MSAs generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=0.05 (MSA 4), ProPIP with k=0.10 (MSA 5), ProPIP with k=0.25 (MSA 6), ProPIP with k=0.50 (MSA 7), ProPIP with k=1 (MSA 8), and ProPIP with k=2 (MSA 9). Note: Black represents Characters and White represents Indels.

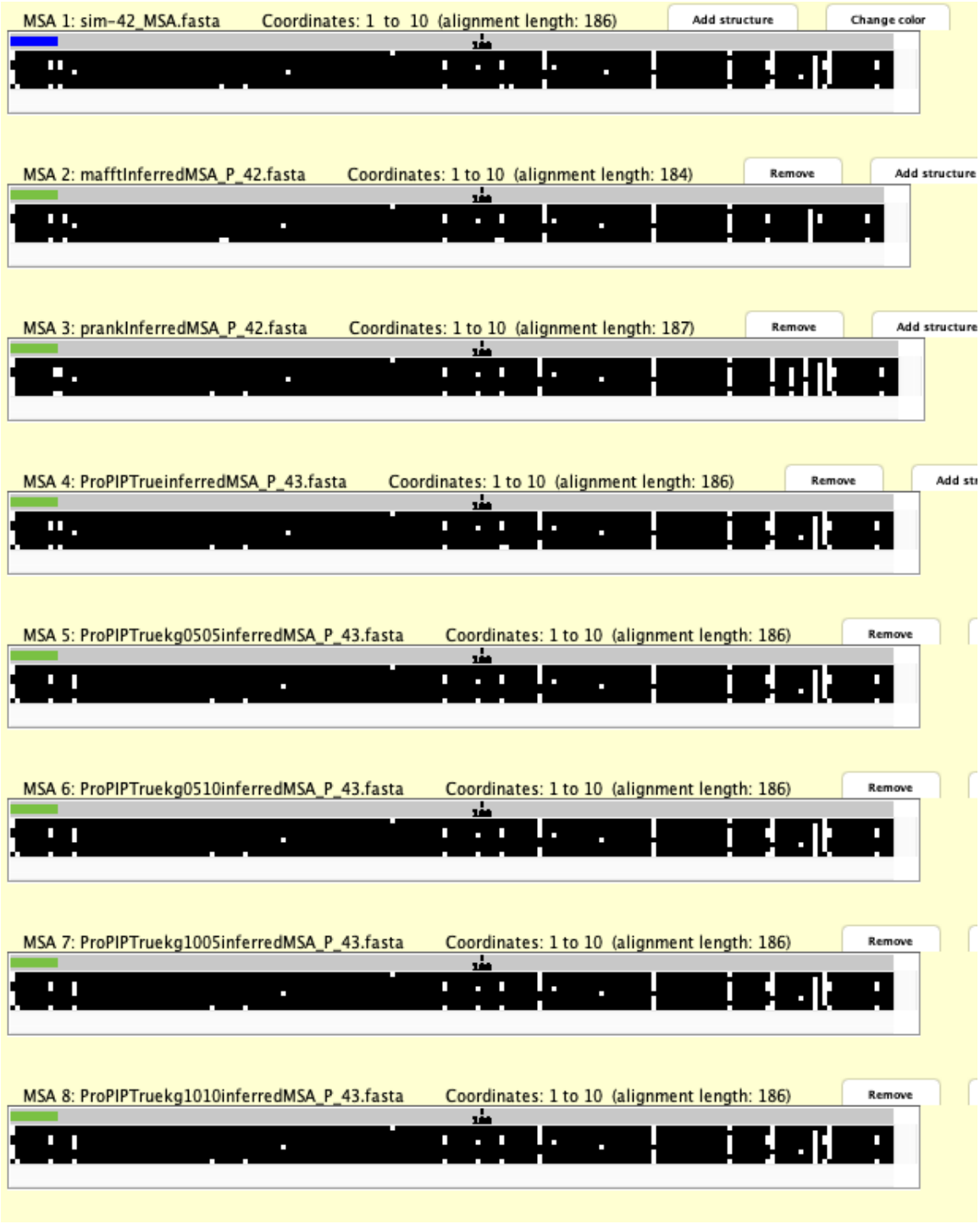


Figure G.8: The Pixel Plot[1] (Section5.5)). A simulated True MSA using PIPJava is compared with MSAs generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), ProPIP with k=0.05 under Gamma n=4, α=0.05 (MSA 5), ProPIP with k=0.05 under Gamma n=4, α=0.10 (MSA 6), ProPIP with k=0.10 under Gamma n=4, α=0.05 (MSA 7), and ProPIP with k=0.10 under Gamma n=4, α=0.10 (MSA 8), Note: Black represents Characters and White represents Indels.

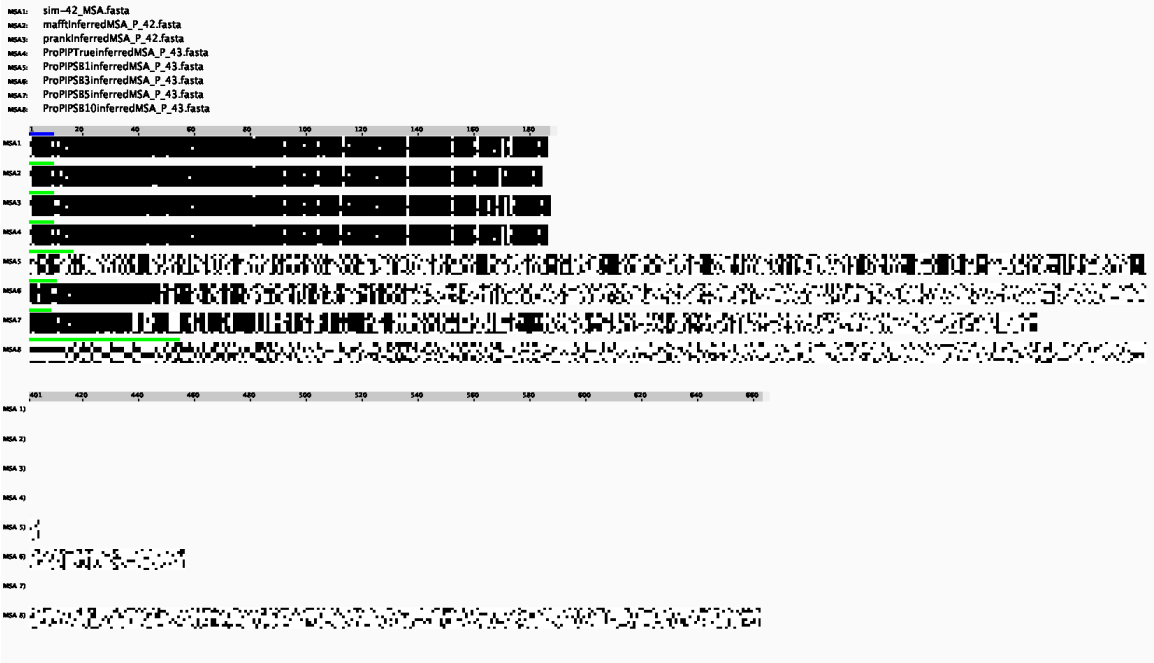


Figure G.9: The Pixel Plot[1] (Section5.5)). A simulated True MSA using PIPJava is compared with MSAs generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), ProPIP with k=1 under SBDP with T= 1 (MSA 5), ProPIP with k=1 under SBDP with T= 3 (MSA 6), ProPIP with k=1 under SBDP with T= 5 (MSA 7), and ProPIP with k=1 under SBDP with T= 10 (MSA 8). Note: Black represents Characters and White represents Indels.

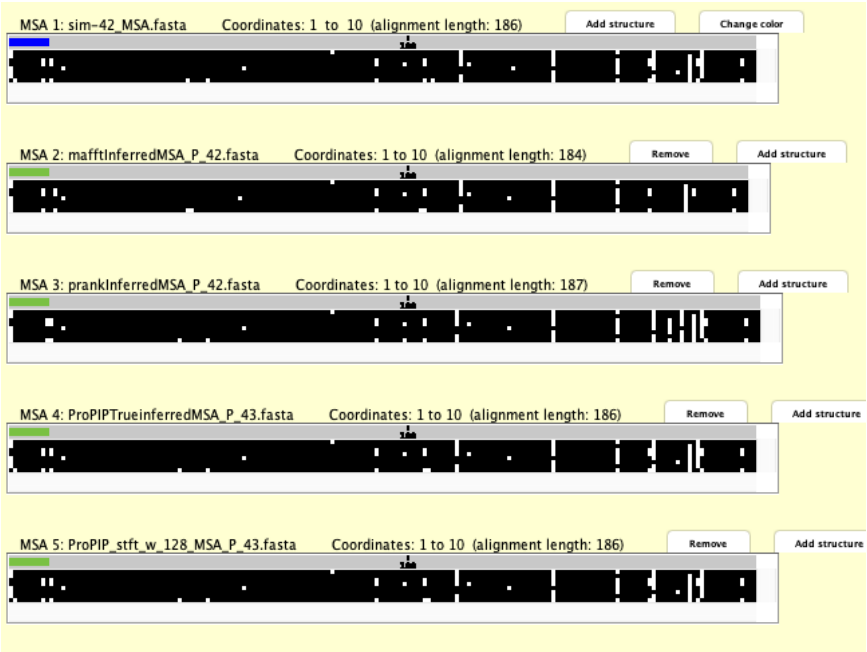


Figure G.10: The Pixel Plot[1] (Section5.5)). A simulated True MSA using PIPJava is compared with MSAs generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), and ProPIP with k=1 under STFT with fiter: welch and filter size: 128 (MSA 5). Note: Black represents Characters and White represents Indels.

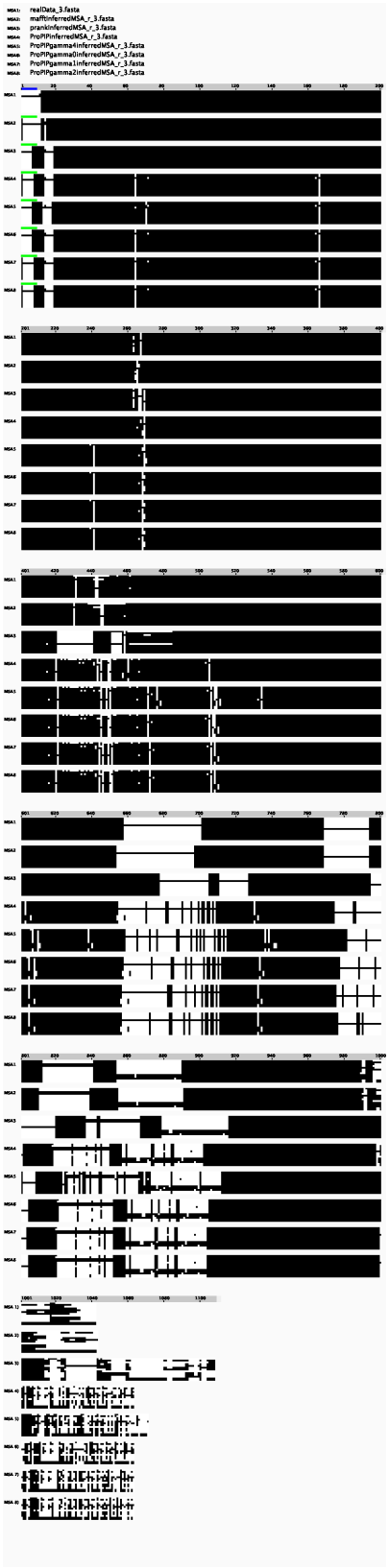


Figure G.11: The Pixel Plot[1] (Section5.5)). A protein alignment (MSA 1 Prot_atpB) from the study is compared with MSAs generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), ProPIP with k=1 under Gamma n=4, α=0.10 (MSA 5), ProPIP with k=1 under Gamma n=4, α=0.50 (MSA 6), ProPIP with k=1 under Gamma n=4, α=1 (MSA 7), ProPIP with k=1 under Gamma n=4, α=2 (MSA 8), and ProPIP with k=1 under Gamma n=4, α=3 (MSA 9). Note: Black represents Characters and White represents Indels.

Figure G.12: The Pixel Plot[1] (Section5.5)). A protein alignment (MSA 1 Prot_atpB) from the study is compared with MSAs generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=0.05 (MSA 4), ProPIP with k=0.10 (MSA 5), ProPIP with k=0.25 (MSA 6), ProPIP with k=0.50 (MSA 7), ProPIP with k=1 (MSA 8), and ProPIP with k=2 (MSA 9). Note: Black represents Characters and White represents Indels.

Figure G.13: The Pixel Plot[1] (Section5.5)). A protein alignment (MSA 1 Prot_atpB) from the study is compared with MSAs generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), ProPIP with k=0.05 under Gamma n=4, α=0.05 (MSA 5), ProPIP with k=0.05 under Gamma n=4, α=0.10 (MSA 6), ProPIP with k=0.10 under Gamma n=4, α=0.05 (MSA 7), and ProPIP with k=0.10 under Gamma n=4, α=0.10 (MSA 8), Note: Black represents Characters and White represents Indels.

Figure G.14: The Pixel Plot[1] (Section5.5)). A protein alignment (MSA 1 Prot_atpB) from the study is compared with MSAs generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=1 (MSA 4), and ProPIP with k=1 under STFT with fiter: welch and filter size: 128 (MSA 5). Note: Black represents Characters and White represents Indels.

Poster

Zürcher Hochschule für Angewandte Wissenschaften

zh aw

A study of dynamics of Indels using ProPIP, PRANK and MAFFT

Student: Eldhose Poulose, Master's Thesis 2020
Supervisors: Dr. Manuel Gil, Dr. Maria Anisimova, Dr. Massimo Maiolo

ACGT Department, Institute of Applied Simulation (IAS), ZHAW, Campus Reidbach, 8820 Wädenswil, Switzerland

Abstract

Evolutionary changes happens over time in the genomes of species. Two of the most important mechanisms of evolutionary changes are substitutions and indels (insertions and deletions). Analyses of genomic sequences typically rely on multiple sequence alignments (MSA), which infer indels. The Applied Computational Genomics Team has developed an MSA estimator (ProPIP). It relies on an explicit evolutionary model of indel (as opposed to more traditional aligners) termed Poisson Indel Process (PIP).

In this thesis we analyse and compare the MSAs inferred by ProPIP and two state of art aligners PRANK and MAFFT . We use simulated as well as real data in this study. In particular, we are interested in the suitability of PIP (which is a single character indel model) to infer long indels. We also examine the location of the indel events on the phylogenies implied by the different aligners.

MSA Evaluation Methods

Main Results

	True(id)	MAFFT v7.453		PRANK v.170427		ProPIP	
(100,8)		id	Xid	id	Xid	id	Xid
nIndels	11511	9279	8971	10226	9692	19539	14418
Max-IL	37	45	45	58	38	30	33
Mean	3.116	3.054	3.780	3.306	3.488	2.041	2.767
Median	1	2	2	2	2	1	1
SD	3.850	4.394	4.506	3.307	3.489	2.132	3.321

Table 6.1: The summary statistics of the 'true' Indel length distribution of INDELible data (True(id)) is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by MAFFT v7.453, PRANK v.170427, and ProPIP. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution

In particular, we are interested in the suitability of PIP (which is a single character indel model) to infer long indels. We also examine the location of the indel events on the phylogenies implied by the different aligners.

Figure 6.2: The Pixel Plot[1] (Section 5.5). A simulated 'true' MSA using INDELible is compared with MSA's generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), and ProPIP (MSA 4). Note: black pixel represents Characters and white pixel represents Indels.

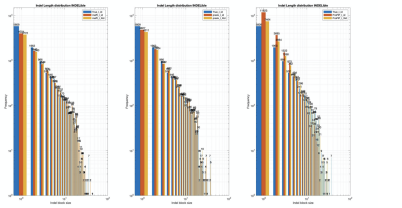


Figure 6.3: The Log-Log Plot. The 'true' Indel length distribution of INDELible data is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by MAFFT v7.453, PRANK v.170427, and ProPIP. Note: The legend blue represents 'true' Indel length distribution, red represents inferred indel length distribution and yellow represents the inferred indel block distribution.

	True	k0.05		k0.10		k0.25		k0.50		k2		k3	
(100,8)	(id)	id	Xid	id	Xid	id	Xid	id	Xid	id	Xid	id	Xid
nIndels	11511	13725	10466	13745	10618	14562	11060	16030	12231	25071	17307	30713	20091
Max-IL	37	30	30	30	30	30	30	30	32	43	105	37	118
Mean	3.116	2.119	2.779	2.161	2.797	2.140	2.817	2.102	2.755	2.090	3.028	2.1663	3.312
Median	1	1	1	1	1	1	1	1	1	1	2	1	2
SD	3.850	2.207	3.316	2.261	3.352	2.226	3.376	2.255	3.294	2.230	3.985	2.347	4.740

Table 6.3: The summary statistics of the 'true' Indel length distribution of INDELible data (True(id)) is compared with Indel length and Indel block distribution statistics (See Section 5.2 and 5.3) generated by ProPIP with k=0.05, ProPIP with k=0.10, ProPIP with k=0.25, ProPIP with k=0.50, ProPIP with k=2, ProPIP with k=3. Note: The 'id' represents indel length distribution and 'Xid' represents indel block distribution.

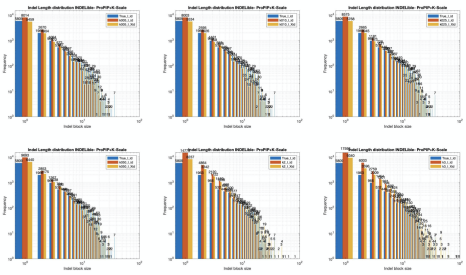


Figure 6.6: The Log-Log Plot. The 'true' Indel length distribution of INDELible data is compared with Indel length and Indel block distribution (See Section 5.2 and 5.3) generated by (from top-left to bottom-right) ProPIP with k=0.05, ProPIP with k=0.10, ProPIP with k=0.25, ProPIP with k=0.50, ProPIP with k=2, and ProPIP with k=3. Note: The legend blue represents 'true' Indel length distribution, red represents inferred indel length distribution and yellow represents the inferred indel block distribution.

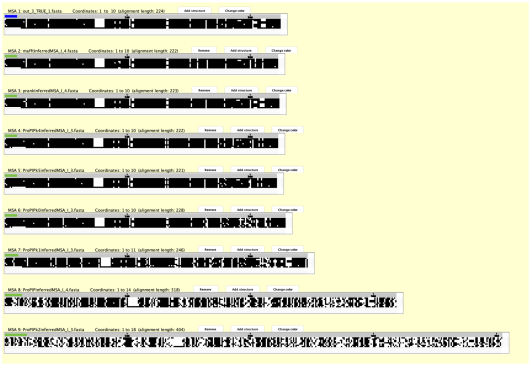


Figure 6.5: The Pixel Plot[1] (Section 5.5). A simulated 'true' MSA using INDELible is compared with MSA's generated by MAFFT v7.453 (MSA 2), PRANK v.170427 (MSA 3), ProPIP with k=0.05 (MSA 4), ProPIP with k=0.10 (MSA 5), ProPIP with k=0.25 (MSA 6), ProPIP with k=0.50 (MSA 7), ProPIP with k=1 (MSA 8), and ProPIP with k=2 (MSA 9). Note: black pixel represents Characters and white pixel represents Indels.

Conclusion and Future works

References

1. M.Maiolo, X.Zhang, M.Gil, and M.Anisimova. Progressive multiple sequence alignment with indel evolution.BMC Bioinformatics, 19(1):1–8, 2018.
2. M.Maiolo Dissertation, https://serval.unil.ch/en/notice/serval:BIBD24577D3A885, 2019.
3. L.Gatti M.Maiolo, ProPIP, castor aligner, manual, https://github.com/acg-team/castoraligner, 2017.
4. W.Fletcher and Z.Yang, INDELible:A flexible simulator of biological sequence evolution.Molecular Biology and Evolution, 26(8):1879–1888, 2009
5. C.L.Anderson, C.L.Strope, and E.N.Moriyama. SuiteMSA:Visual tools for multiplesequence alignment comparison and molecular sequence simulation.BMC Bioinformatics,

Declaration of Originality

Master's Thesis for the School of Life Sciences and Facility Management

By submitting this Master's thesis, the student attests to the fact that all the work included in the assignment is their own and was written without the help of a third party.

The student declares that all sources in the text (including Internet pages) and appendices have been correctly disclosed. This means that there has been no plagiarism, i.e. no sections have been partially or wholly taken from other texts and represented as the student's own work or included without being correctly referenced.

Any misconduct will be dealt with according to paragraphs 39 and 40 of the General Academic Regulations for Bachelor's and Master's Degree Courses at the Zurich University of Applied Sciences (dated 29 January 2008) and subject to the provisions for disciplinary action stipulated in the University regulations (Rahmenprüfungsordnung ZHAW (RPO)).

Town/City, Date:

.....

Signature:

.....

Erklärung betreffend Einwilligung zur elektronischen Veröffentlichung einer Masterarbeit auf der ZHAW Digitalcollection

Ich erkläre mich damit einverstanden, dass meine Arbeit elektronisch gespeichert und in der ZHAW Digital-collection der ZHAW Hochschulbibliothek öffentlich zugänglich gemacht wird. Das Recht, die Arbeit an anderer Stelle zu veröffentlichen, wird durch diese Erklärung grundsätzlich nicht berührt. Ich bin damit einverstanden, dass die Arbeit, namentlich zum Zweck der Archivierung, in andere Dateiformate konvertiert oder anderweitig technisch verändert wird.

Ich versichere, dass der Veröffentlichung der Arbeit keine Rechte Dritter, insbesondere in Bezug auf im Werk enthaltenen Abbildungen, entgegenstehen.

Ort, Datum: .....

Unterschrift: .....

Titel der Arbeit: A study of dynamics of Indels using ProPIP, PRANK and MAFFT

Name der/des Studierenden: Eldhose Poulose

Name der/des 1.Korrigierenden: Dr. Manuel Gil

Welche Schlagwörter schlagen Sie für die öffentliche online Suche vor? Progressive sequence alignment, ProPIP, Poisson Indel Process, INDELible, PRANK , MAFFT, Rate Variation Across Sites (ASRV), Stochastic backtracking DP algorithm (SBDP), The Pixel Plot.