SIMULATIONS AND DATA STRUCTURES TO STUDY DRUG RESISTANCE

Dissertation

zur Erlangung des Grades
des Doktors der Naturwissenschaften
der Fakultät für Mathematik und Informatik
der Universität des Saarlandes

vorgelegt von

Sanjay Kumar Srikakulam

Saarbrücken,

2024

Kolloqium: 03.12.2024

Dekan der Fakultät: Prof. Dr. Roland Speicher

Komitee:
Vorsitzender: Prof. Dr. Sven Rahmann
Gutachterin: Prof. Dr. Olga V. Kalinina
Gutachter: Prof. Dr. Volkhard Helms
Beisitzer: Dr. Alexander Gress

ABSTRACT

Drug resistance is a critical health issue that significantly impacts the success of treatments across various diseases, often leading to relapse and treatment failure. Over the recent decades, the emergence of drug resistance has posed a significant barrier to effective therapeutic interventions. It can develop due to several factors, most notably through mutations in the genetic makeup of organisms or individual cells, which subsequently alter the structure and function of key proteins. Understanding and addressing drug resistance requires a comprehensive approach that combines advanced computational techniques with molecular insights.

In this thesis, we explore the impact of mutations by using modern computational techniques to handle this challenge. The first project uses molecular dynamics (MD) simulations to study how specific mutations influence protein dynamics, structure, and function. Examining the mutation-induced structural changes in proteins helps us understand how mutations may decrease drug efficacy, which is particularly relevant in diseases like cancer and viral infections. We explore two case studies: the receptor tyrosine kinase KIT, where post-translational modifications and mutations stabilize the active conformation, leading to consecutive activation of the protein and reducing drug sensitivity in cancers, and the NS3 protease in hepatitis C virus (HCV), where interactions between mutations stabilize the protein structure, maintaining resistance to direct-acting antiviral agents. These findings enhance our understanding of drug resistance mechanisms and aid in developing more effective therapeutic strategies.

To complement these molecular insights, the second project introduces MetaProFi, a novel state-of-the-art computational tool developed to enhance large-scale genomic analysis. MetaProFi addresses the limitations of traditional sequence analysis methods like BLAST, particularly when dealing with large datasets. MetaProFi is unique in its ability to index both nucleotide and amino acid sequences and offers the additional feature of querying amino acid indexes using nucleotide sequences. This feature is crucial for identifying functionally relevant genetic variants

with greater accuracy, as protein sequences tend to be more conserved across evolutionary distances. MetaProFi implements advanced optimizations like data chunking, shared memory systems, and compression algorithms and offers a scalable and efficient solution for large-scale genomic analysis. This integration of molecular dynamics insights with large-scale genomic analysis allows us to explore the genetic basis of drug resistance more effectively, offering a comprehensive toolkit for developing targeted and personalized therapeutic strategies.

ZUSAMMENFASSUNG

Arzneimittelresistenz ist ein kritisches medizinischesproblem, das den Erfolg von Behandlungen bei verschiedenen Krankheiten erheblich beeinträchtigt und häufig zu Rückfällen und Behandlungsversagen führt. In den letzten Jahrzehnten hat das Auftreten von Arzneimittelresistenzen ein erhebliches Hindernis für wirksame therapeutische Interventionen dargestellt. Sie kann durch verschiedene Faktoren entstehen, vor allem durch Mutationen im Erbgut von Organismen oder einzelnen Zellen, die in der Folge die Struktur und Funktion von Schlüsselproteinen verändern. Um Arzneimittelresistenzen zu verstehen und zu bekämpfen, ist ein umfassender Ansatz erforderlich, der fortgeschrittene Computertechniken mit molekularen Erkenntnissen kombiniert.

In dieser Arbeit erforschen wir die Auswirkungen von Mutationen mit Hilfe moderner Computertechniken, um diese Herausforderung zu bewältigen. Das erste Projekt nutzt Simulationen der Molekulardynamik (MD), um zu untersuchen, wie spezifische Mutationen die Dynamik, Struktur und Funktion von Proteinen beeinflussen. Die Untersuchung der mutationsbedingten strukturellen Veränderungen in Proteinen hilft uns zu verstehen, wie Mutationen die Wirksamkeit von Medikamenten verringern können, was bei Krankheiten wie Krebs und Virusinfektionen besonders wichtig ist. Wir untersuchen zwei Fallstudien: die Rezeptortyrosinkinase KIT, bei der posttranslationale Modifikationen und Mutationen die aktive Konformation stabilisieren, was zu einer konsekutiven Aktivierung des Proteins und einer geringeren Empfindlichkeit gegenüber Arzneimitteln bei Krebserkrankungen führt, und die NS3-Protease des hepatitis-C-Virus (HCV), bei der Wechselwirkungen zwischen Mutationen die Proteinstruktur stabilisieren und die Resistenz gegenüber direkt wirkenden antiviralen Mitteln aufrechterhalten. Diese Erkenntnisse verbessern unser Verständnis der Mechanismen der Arzneimittelresistenz und helfen bei der Entwicklung wirksamerer therapeutischer Strategien.

Um diese molekularen Erkenntnisse zu ergänzen, wird im Rahmen des zweiten Projekts MetaProFi eingeführt, ein neuartiges, hochmoder-

nes Computerprogramm, das zur Verbesserung der Genomanalyse in großem Maßstab entwickelt wurde. MetaProFi überwindet die Grenzen herkömmlicher Sequenzanalysemethoden wie BLAST, insbesondere bei großen Datensätzen. MetaProFi ist einzigartig in seiner Fähigkeit, sowohl Nukleotid- als auch Aminosäuresequenzen zu indizieren und bietet zusätzlich die Möglichkeit, Aminosäureindizes anhand von Nukleotidsequenzen abzufragen. Diese Funktion ist entscheidend für die Identifizierung funktionell relevanter genetischer Varianten mit größerer Genauigkeit, da Proteinsequenzen dazu neigen, über evolutionäre Distanzen hinweg besser konserviert zu sein. MetaProFi implementiert fortschrittliche Optimierungen wie Data Chunking, Shared-Memory-Systeme und Kompressionsalgorithmen und bietet eine skalierbare und effiziente Lösung für groß angelegte Genomanalysen. Diese Integration von Erkenntnissen aus der Molekulardynamik mit groß angelegten Genomanalysen ermöglicht es uns, die genetischen Grundlagen der Arzneimittelresistenz effektiver zu erforschen und bietet ein umfassendes Instrumentarium für die Entwicklung gezielter und personalisierter therapeutischer Strategien.

ACKNOWLEDGMENTS

I want to express my deepest gratitude to Prof. Dr. Olga Kalinina for allowing me to pursue my doctoral studies and for her unwavering support. I am incredibly thankful for her trust in me as the first employee to move with her to the Helmholtz Institute for Pharmaceutical Research Saarland (HIPS) and for allowing me to establish and manage the bioinformatics HPC core facility. Her guidance has greatly nurtured my professional and personal growth, making my time in the group genuinely memorable. The fun barbecues at her place also added a special flavor to this journey.

I would like thank Prof. Dr. Volkhard Helms for taking the time to review my thesis.

I sincerely thank Prof. Dr. Sven Rahmann for his valuable discussion and insights on optimizing MetaProFi. I am also profoundly grateful to Dr. Tomas Bastys for his support during the molecular dynamics simulations. His technical expertise and willingness to help have been invaluable to this work. I would also like to express my heartfelt thanks to Prof. Dr. Dr. Robert Bals. Working with him at Saarland University Hospital as a Bioinformatician has been an exceptional experience.

My heartfelt appreciation goes to my incredible colleagues and friends, Alexander, Sebastian, and Fawaz. Our lunches at the Mensa and Philo Café, followed by our after-lunch sweet cravings at iCoffee, and our countless discussions on numerous topics made this journey one of my life's best and sweetest experiences. I am also grateful to Amay, Carla, Chen, Alper, and Roman from our group, as well as to Mark and Michael from HIPS IT for their support and assistance.

Special thanks to Alex and Sebastian for proofreading parts of the thesis.

I am also thankful to the Galaxy Freiburg team, particularly Dr. Björn Grüning, for his support during the writing of this thesis and for allowing me to embark on the next chapter of my career. I am grateful to my team friends, Pavan, Deepti, Anup, Mira, Paul, Saim, Sebastian, and Manuel.

I am deeply grateful to my friends Shashwat Sahay and Sebastian Wendland. Shashwat has been a terrific friend since 2011, always there to support me. Sebastian, who gave me my first job at DFKI, has become an incredible friend, and his mentorship has been instrumental in my personal and professional development. I also want to thank Jennifer, Paul, and several others from DFKI and the computer graphics chair at Saarland University for all the fun times.

Lastly, but most importantly, I cannot find enough words to thank my parents, my brother, and his family. Their love, support, and belief in me have been the cornerstone of my life, shaping me into who I am today and keeping me going. My brother has been my rock, supporting me through every phase of my life. Thank you from the bottom of my heart!

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1

INTRODUCTION

Life on Earth is believed to have originated approximately 3.5 to 4 billion years ago [1–3], starting as single-celled organisms that eventually evolved into complex multicellular life forms. The intricate interplay of genes, proteins, and regulatory networks has driven this transition. According to the central dogma of molecular biology [4], DNA transcribes to RNA, and RNA translates to protein, directing the flow of genetic information within cells. Bioinformatics and computational biology have emerged as essential fields in molecular biology, providing tools for analyzing and interpreting the vast amounts of data generated by today's research. The dawn of sequencing technologies, especially next-generation sequencing (NGS) technologies, has transformed our understanding of biological systems, leading to various OMICS disciplines focusing on comprehensive datasets characterizing different biomolecular components. For example, genomics studies the complete DNA within an organism, transcriptomics examines the RNA transcripts, proteomics studies the entire proteins, and metabolomics explores the metabolites within a biological sample. These fields collectively provide insights into the mechanisms of gene expression, protein function, and metabolic pathways.

Further advancements in these fields led to expansion into more specialized areas. This evolution has given rise to fields such as epigenetics, which studies the chemical modifications of DNA and histones that regulate gene expression; interactomics, which focuses on the complex networks of protein interactions; and metagenomics, which explores DNA from environmental samples. Bioinformatics and computational biology play a crucial role in managing, processing, and analyzing the unprecedented volumes of data produced by these diverse OMICS fields; they also aid in facilitating the transformation and integration of diverse data types, enabling the comprehensive mapping of information from different fields and supporting the simulation of entire biological systems. Advances in sequencing technologies and large-scale initiatives

have significantly enhanced our understanding of genetic and phenotypic diversity, with the Human Genome Project (HGP) [5] being a landmark initiative in this regard.

Due to these advancements, we are experiencing overwhelming growth of sequence data in public databases like the European Nucleotide Archive (ENA) [6] and Sequence Read Archive (SRA) [7]. As the size of the databases reached the petabyte scale, it has become difficult to support online searches in these databases using classical tools like Basic Local Alignment Search Tool (BLAST) [8]. This calls for a paradigm shift in designing and developing novel tools to tackle such datasets. This rapid growth creates a significant demand for effective management, and the need for tools to process, store, and query these extensive collections of sequence data without high memory and storage requirements constitutes a major computational challenge. Analyzing these abundant data will lead to opportunities for great scientific discoveries, including identifying novel genetic variants and mutations that affect protein function, which can provide insights into disease mechanisms, enhance drug development, and uncover new therapeutic targets.

On the other end, proteins are fundamental to cellular function and are crucial in this context. The protein's primary structure, which is its linear sequence of amino acids, fundamentally determines its higher-order structures, as demonstrated by Christian Anfinsen in 1973 [9]. Techniques of structural biology, including X-ray crystallography, nuclear magnetic resonance (NMR) spectroscopy, and cryo-electron microscopy (cryo-EM), have provided crucial insights into the three-dimensional structures of proteins, revealing how these structures influence their biological functions. Therefore, understanding protein structures and dynamics is essential for deciphering their roles in biological processes and disease mechanisms. Mutations in DNA sequences can lead to changes in protein structures, potentially resulting in a change in protein function, leading, among other consequences, to drug resistance. For example, mutations can affect protein folding and stability or affinity towards a ligand, leading to resistance against therapeutics. To address these challenges, molecular dynamics (MD) simulations have been a vital tool for studying protein dynamics at the atomistic level. MD simulations (Chapter 2.8) provide insights into how proteins undergo conformational

changes and interact with ligands, which is crucial for drug design and understanding resistance mechanisms.

In this thesis, we present two case studies (Chapter 3) that utilize MD simulations to explore protein dynamics and the effects of post-translational modifications and mutations on protein structures. The first study investigates the phosphorylation at Y823 and the Y823D mutation in the KIT receptor tyrosine kinase [10] (Chapter 3.2), which is crucial for cell differentiation, proliferation, and survival. Dysregulation of KIT, often due to such modifications or mutations, is associated with various cancers, including gastrointestinal stromal tumors, leukemia, and melanoma [11–13]. Our study demonstrates how phosphorylation at Y823 stabilizes the protein's active state while the Y823D mutation similarly shifts the dynamic equilibrium, both contributing to drug resistance. The second study examines the NS3-Q80K variant in the NS3-4A protease of the Hepatitis C virus (HCV) [14] (Chapter 3.3), known for its association with treatment failure of direct-acting antiviral agents, destabilizing the protease's fold. Despite this destabilization, the NS3-Q80K variant maintains replicative fitness comparable to the wild-type (WT) virus. This is achieved through epistatic interactions with other amino acid substitutions, which stabilize the protein fold without enhancing enzymatic activity or replicative fitness. MD simulations show that these epistatic interactions increase the total number of residue contacts, thereby compensating for protein folding instability.

Following this, we introduce a novel state-of-the-art tool, MetaProFi [15] (Chapter 4) that can efficiently index protein and nucleotide sequence data with reduced computing requirements by combining efficient probabilistic data structures with advanced computational methods like optimized data chunking, shared memory systems, and compression algorithms. MetaProFi indexes both nucleotide and amino acid sequences and offers the feature to query an amino acid database (index) using nucleotide sequences as input. By enabling this, MetaProFi aims to bridge the gap in sequence analysis, making it possible to explore genetic variants with precision and efficiency and focusing directly on those that impact protein function.

1.1 FIRST AUTHOR PUBLICATIONS RELATED TO DOCTORAL STUDIES

Sanjay K. Srikakulam, Sebastian Keller, Fawaz Dabbaghie, Robert Bals, Olga V Kalinina, **"MetaProFi: an ultrafast chunked Bloom filter for storing and querying protein and nucleotide sequence data for accurate identification of functionally relevant genetic variants"**, *Bioinformatics*, Volume 39, Issue 3, March 2023, btad101

Abstract: Bloom filters are a popular data structure that allows rapid searches in large sequence datasets. So far, all tools work with nucleotide sequences; however, protein sequences are conserved over longer evolutionary distances, and only mutations on the protein level may have any functional significance. We present MetaProFi, a Bloom filter-based tool that, for the first time, offers the functionality to build indexes of amino acid sequences and query them with both amino acid and nucleotide sequences, thus bringing sequence comparison to the biologically relevant protein level. MetaProFi implements additional efficient engineering solutions, such as a shared memory system, chunked data storage, and efficient compression. In addition to its conceptual novelty, MetaProFi demonstrates state-of-the-art performance and excellent memory consumption-to-speed ratio when applied to various large datasets.

Georg Dultz[†], Sanjay K. Srikakulam[†], Michael Konetschnik[†], Tetsuro Shimakami, Nadezhda T. Doncheva, Julia Dietz, Christoph Sarrazin, Ricardo M. Biondi, Stefan Zeuzem, Robert Tampé, Olga V. Kalinina, Christoph Welsch, **"Epistatic interactions promote persistence of NS3-Q80K in HCV infection by compensating for protein folding instability"**, *Journal of Biological Chemistry*, 297(3), 101031.

Abstract: The Q80K polymorphism in the NS3-4A protease of the hepatitis C virus is associated with treatment failure of direct-acting antiviral agents. This polymorphism is highly prevalent in genotype 1a infections and stably transmitted between hosts. Here, we investigated the underlying molecular mechanisms of evolutionarily conserved coevolving amino acids in NS3-Q80K and revealed potential implications of epistatic interactions in immune escape and variants persistence. Using

[†]These authors contributed equally to this work.

purified protein, we characterized the impact of epistatic amino acid substitutions on the physicochemical properties and peptide cleavage kinetics of the NS3-Q80K protease. We found that Q80K destabilized the protease protein fold ($p < 0.0001$). Although NS3-Q80K showed reduced peptide substrate turnover ($p < 0.0002$), replicative fitness in an H77S.3 cell culture model of infection was not significantly inferior to the WT virus. Epistatic substitutions at residues 91 and 174 in NS3-Q80K stabilized the protein fold ($p < 0.0001$) and leveraged the WT protease stability. However, changes in protease stability inversely correlated with enzymatic activity. In infectious cell culture, these secondary substitutions were not associated with a gain of replicative fitness in NS3-Q80K variants. Using molecular dynamics, we observed that the total number of residue contacts in NS3-Q80K mutants correlated with protein folding stability. Changes in the number of contacts reflected the compensatory effect on protein folding instability by epistatic substitutions. In summary, epistatic substitutions in NS3-Q80K contribute to viral fitness by mechanisms not directly related to RNA replication. By compensating for protein-folding instability, epistatic interactions likely protect NS3-Q80K variants from immune cell recognition.

Sanjay K. Srikakulam, Tomas Bastys, Olga V. Kalinina, **"A shift of dynamic equilibrium between the KIT active and inactive states causes drug resistance"**, *Proteins*. 2020; 88: 1434–1446.

Abstract: Tyrosine phosphorylation, a highly regulated post-translational modification, is carried out by the enzyme tyrosine kinase (TK). TKs are important mediators in signaling cascades, facilitating diverse biological processes in response to stimuli. TKs may acquire mutations leading to malignancy and are viable targets for anti-cancer drugs. Mast/stem cell growth factor receptor KIT is a TK involved in cell differentiation, whose dysregulation leads to various types of cancer, including gastrointestinal stromal tumors, leukemia, and melanoma. KIT can be targeted by a range of inhibitors that predominantly bind to the inactive state of the enzyme. A mutation Y823D in the activation loop of KIT is known to be responsible for the loss of sensitivity to some drugs in metastatic tumors. We used all-atom molecular dynamics simulations to study the impact of Y823D on the KIT conformation and dynamics and compared it to the effect of phosphorylation of Y823. We simulated in total 6.4 μs of

wild-type, mutant and phosphorylated KIT in the active- and inactive-state conformations. We found that Y823D affects the protein dynamics differently: in the active state, the mutation increases the protein stability, whereas in the inactive state it induces local destabilization, thus shifting the dynamic equilibrium towards the active state, altering the communication between distant regulatory regions. The observed dynamics of the Y823D mutant is similar to the dynamics of KIT phosphorylated at position Y823, thus we hypothesize that this mutation mimics a constitutively active kinase, which is not responsive to inhibitors that bind its inactive conformation.

1.2 COAUTHOR PUBLICATIONS DURING DOCTORAL STUDIES

Fawaz Dabbaghie, Sanjay K. Srikakulam, Tobias Marschall, Olga V Kalinina, **"PanPA: generation and alignment of panproteome graphs"**, *Bioinformatics Advances*, Volume 3, Issue 1, 2023, vbad167.

Abstract: Compared to eukaryotes, prokaryote genomes are more diverse through different mechanisms, including a higher mutation rate and horizontal gene transfer. Therefore, using a linear representative reference can cause a reference bias. Graph-based pangenome methods have been developed to tackle this problem. However, comparisons in DNA space are still challenging due to this high diversity. In contrast, amino acid sequences have higher similarity due to evolutionary constraints, whereby a single amino acid may be encoded by several synonymous codons. Coding regions cover the majority of the genome in prokaryotes. Thus, panproteomes present an attractive alternative leveraging the higher sequence similarity while not losing much of the genome in non-coding regions. We present PanPA, a method that takes a set of multiple sequence alignments of protein sequences, indexes them, and builds a graph for each multiple sequence alignment. In the querying step, it can align DNA or amino acid sequences back to these graphs. We first showcase that PanPA generates correct alignments on a panproteome from 1350 *Escherichia coli*. To demonstrate that panproteomes allow comparisons at longer phylogenetic distances, we compare DNA and protein alignments from 1073 *Salmonella enterica* assemblies against *E.coli* reference genome, pangenome, and panproteome using BWA, GraphAligner, and PanPA, respectively; with PanPA aligning around

22% more sequences. We also aligned a DNA short-reads whole genome sequencing (WGS) sample from *S.enterica* against the *E.coli* reference with BWA and the panproteome with PanPA, where PanPA was able to find alignment for 68% of the reads compared to 5% with BWA.
My contribution: I contributed to the part of the code development for the tool PanPA.

Alexander Gress, Sanjay K. Srikakulam, Sebastian Keller, Vasily Ramensky, Olga V Kalinina, **"d-StructMAn: Containerized structural annotation on the scale from genetic variants to whole proteomes"**, *GigaScience*, Volume 11, 2022, giac086.
Abstract: Structural annotation of genetic variants in the context of intermolecular interactions and protein stability can shed light onto mechanisms of disease-related phenotypes. Three-dimensional structures of related proteins in complexes with other proteins, nucleic acids, or ligands enrich such functional interpretation, since intermolecular interactions are well conserved in evolution. We present d-StructMAn, a novel computational method that enables structural annotation of local genetic variants, such as single-nucleotide variants and in-frame indels, and implements it in a highly efficient and user-friendly tool provided as a Docker container. Using d-StructMAn, we annotated several very large sets of human genetic variants, including all variants from ClinVar and all amino acid positions in the human proteome. We were able to provide annotation for more than 46% of positions in the human proteome representing over 60% proteins. d-StructMAn is the first of its kind and a highly efficient tool for structural annotation of protein-coding genetic variation in the context of observed and potential intermolecular interactions. d-StructMAn is readily applicable to proteome-scale datasets and can be an instrumental building machine-learning tool for predicting genotype-to-phenotype relationships.
My contribution: I contributed to the method development and designed and implemented the containerization of the StructMAn application.

Christian Herr, Sebastian Mang, Bahareh Mozafari, Katharina Guenther, Thimoteus Speer, Martina Seibert, Sanjay K. Srikakulam, Christoph Beisswenger, Felix Ritzmann, Andreas Keller, Rolf Mueller, Sigrun Smola,

Dominic Eisinger, Michael Zemlin, Guy Danziger, Thomas Volk, Sabrina Hoersch, Marcin Krawczyk, Frank Lammert, Thomas Adams, Gudrun Wagenpfeil, Michael Kindermann, Constantin Marcu, Zuhair Wolf Dietrich Ataya, Marc Mittag, Konrad Schwarzkopf, Florian Custodis, Daniel Grandt, Harald Schaefer, Kai Eltges, Philipp M Lepper, Robert Bals & On behalf of the CORSAAR Study Group, **"Distinct Patterns of Blood Cytokines Beyond a Cytokine Storm Predict Mortality in COVID-19"**, *Journal of Inflammation Research*, 14, 4651–4667.

Abstract: COVID-19 comprises several severity stages ranging from oligosymptomatic disease to multi-organ failure and fatal outcomes. The mechanisms why COVID-19 is a mild disease in some patients and progresses to a severe multi-organ and often fatal disease with respiratory failure are not known. Biomarkers that predict the course of disease are urgently needed. The aim of this study was to evaluate a large spectrum of established laboratory measurements. Patients from the prospective PULMPOHOM and CORSAAR studies were recruited and comprised 35 patients with COVID-19, 23 with conventional pneumonia, and 28 control patients undergoing elective non-pulmonary surgery. Venous blood was used to measure the serum concentrations of 79 proteins by Luminex multiplex immunoassay technology. Distribution of biomarkers between groups and association with disease severity and outcomes were analyzed. The biomarker profiles between the three groups differed significantly with elevation of specific proteins specific for the respective conditions. Several biomarkers correlated significantly with disease severity and death. Uniform manifold approximation and projection (UMAP) analysis revealed a significant separation of the three disease groups and separated between survivors and deceased patients. Different models were developed to predict mortality based on the baseline measurements of several protein markers. A score combining IL-1ra, IL-8, IL-10, MCP-1, SCF and CA-9 was associated with significantly higher mortality (AUC 0.929). Several newly identified blood markers were significantly increased in patients with severe COVID-19 (AAT, EN-RAGE, myoglobin, SAP, TIMP-1, vWF, decorin) or in patients that died (IL-1ra, IL-8, IL-10, MCP-1, SCF, CA-9). The use of established assay technologies allows for rapid translation into clinical practice.

My contribution: I performed the statistical analysis of the biomarker levels associated with disease severity.

Robert Richter, Mohamed.A.M. Kamal, Mariel A. García-Rivera, Jerome Kaspar, Maximilian Junk, Walid A.M. Elgaher, Sanjay K. Srikakulam, Alexander Gress, Anja Beckmann, Alexander Grißmer, Carola Meier, Michael Vielhaber, Olga Kalinina, Anna K.H. Hirsch, Rolf W. Hartmann, Mark Brönstrup, Nicole Schneider-Daum, Claus-Michael Lehr, **"A hydrogel-based in vitro assay for the fast prediction of antibiotic accumulation in Gram-negative bacteria"**, *Materials Today Bio*, Volume 8, 2020, 100084, ISSN 2590-0064.

Abstract: The pipeline of antibiotics has been for decades on an alarmingly low level. Considering the steadily emerging antibiotic resistance, novel tools are needed for early and easy identification of effective anti-infective compounds. In Gram-negative bacteria, the uptake of anti-infectives is especially limited. We here present a surprisingly simple in vitro model of the Gram-negative bacterial envelope, based on 20% (w/v) potato starch gel, printed on polycarbonate 96-well filter membranes. Rapid permeability measurements across this polysaccharide hydrogel allowed to correctly predict either high or low accumulation for all 16 tested anti-infectives in living *Escherichia coli*. Freeze-fracture TEM supports that the macromolecular network structure of the starch hydrogel may represent a useful surrogate of the Gram-negative bacterial envelope. A random forest analysis of in vitro data revealed molecular mass, minimum projection area, and rigidity as the most critical physicochemical parameters for hydrogel permeability, in agreement with reported structural features needed for uptake into Gram-negative bacteria. Correlating our dataset of 27 antibiotics from different structural classes to reported MIC values of nine clinically relevant pathogens allowed to distinguish active from nonactive compounds based on their low in vitro permeability specifically for Gram-negatives. The model may help to identify poorly permeable antimicrobial candidates before testing them on living bacteria.

My contribution: I implemented and performed the random forest analysis of the in vitro data and wrote R scripts for data analysis.

2

BIOLOGICAL BACKGROUND

2.1 THE ORIGINS AND FUNDAMENTALS

Understanding the origin and evolution of life provides an essential context for studying modern biological systems and developing computational methods to analyze biological data. The Miller-Urey experiment [16] in the 1950s demonstrated that organic compounds, including amino acids, could be synthesized from inorganic precursors under conditions thought to resemble those of the prebiotic Earth. This experiment demonstrated the plausibility that the basic building blocks of life, such as amino acids and nucleic acids, could form spontaneously, supporting the chemical evolution theory [17], which posits that simple organic molecules gradually assemble into complex macromolecules, eventually giving rise to the first living organisms [3, 16, 18].

Among these molecules, RNA is thought to have played a critical role in the early stages of life [2, 3]. The discovery that RNA can catalyze chemical reactions, including the polymerization of nucleotides, supports the idea that the first living systems comprised solely of RNA [2, 19]. The RNA world hypothesis [19] suggests that RNA molecules were pivotal in transitioning from non-living to living matter due to their ability to store genetic information and catalyze biochemical reactions. RNA's dual role in serving as a template for and catalyzing its replication highlights its significance in early molecular evolution [20, 21]. Over time, the interactions between RNA and amino acids led to the development of the more structured genetic code we see today. As evolution progressed, DNA eventually replaced RNA as the primary genetic material due to its greater stability [19]. This transition marked the establishment of the central dogma of molecular biology [4]: the flow of genetic information from DNA to RNA to protein (Figure 2.1).

The evolution of these early molecular systems into fully functional cellular mechanisms illustrates a significant leap in complexity. The simple RNA-based systems initially gave rise to single-celled organisms,

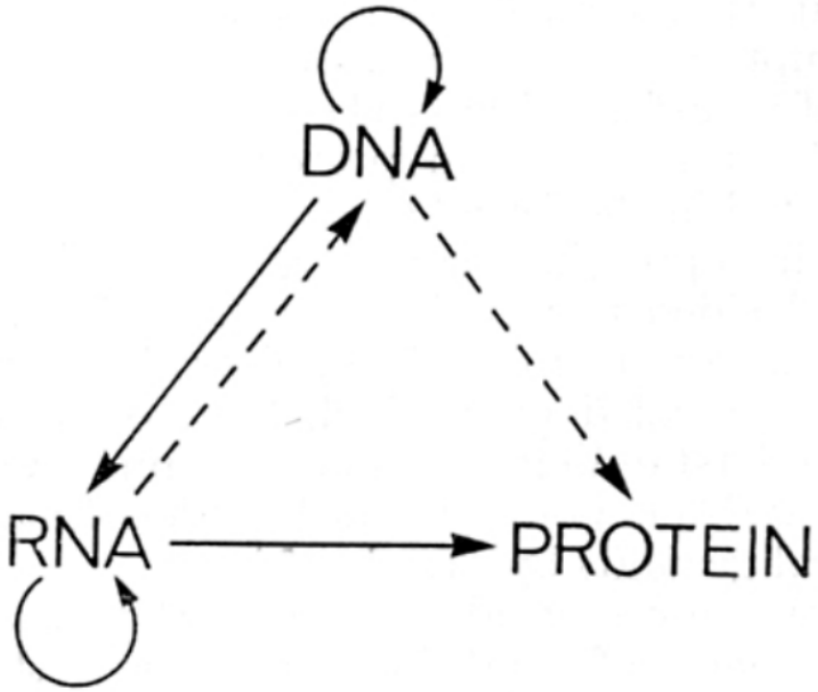

Figure 2.1: *(Source: (Francis Crick, 1970)[4]) Overview of the Central Dogma of Molecular Biology.*

which over time evolved into more complex life forms with multicellular organisms arising on several occasions. The development of prokaryotic and eukaryotic cells marked a pivotal advancement in cellular evolution [1, 3, 22, 23]. The domains [1, 24] of Bacteria and Archaea include prokaryotic cells lacking a nucleus and membrane-bound organelles. Eukaryotic cells, characterized by a nucleus and organelles such as mitochondria and chloroplasts, evolved through endosymbiosis [22, 25, 26] and allowed for the development of complex multicellular organisms. This evolutionary milestone paved the way for the vast life forms observed today.

In today's biology, the central dogma is fundamental to understanding the expression of hereditary information, which involves the intricate processes of replication, transcription, and translation [27]. DNA, or deoxyribonucleic acid, is composed of nucleotides, which are the building blocks containing a phosphate group, a deoxyribose sugar moiety, and one of four nitrogenous bases (each such block is called a nucleotide): adenine (A), thymine (T), guanine (G), and cytosine (C). DNA molecules consist of two strands that form an antiparallel double-helix structure

[28]. The strands are held together by hydrogen bonds between complementary bases: adenine pairs with thymine, and guanine pairs with cytosine. This stable structure ensures accurate replication of genetic material during cell division.

Replication is the process through which a DNA molecule makes an exact copy of itself. This process is crucial for cell division, allowing each new cell to receive a complete set of genetic information. Transcription is the process in which a segment of DNA is copied into RNA (ribonucleic acid). RNA is similar to DNA but differs in a few key aspects: it contains the ribose sugar moiety instead of deoxyribose and uses uracil (U) in place of thymine. RNA is usually single-stranded, which allows it to fold into various structures and perform multiple functions within the cell. There are several types of RNA, with messenger RNA (mRNA) carrying the transcript of the genetic information transferred from DNA to the ribosome, the cellular machinery for protein synthesis. RNA polymerase reads the DNA template during transcription and synthesizes a complementary RNA strand. This strand, known as mRNA, is then transported to the ribosomes. Ribosomes are complex macromolecular machines composed of proteins and ribosomal RNA (rRNA) that facilitate protein synthesis. During translation, mRNA is read in chunks of three bases (codons), each decoded into an amino acid. Peptide bonds link these amino acids to form a polypeptide chain, which eventually folds into a functional protein (Chapter 2.6.2).

Understanding the intricate processes of DNA replication, transcription, and translation has provided deep insights into the molecular mechanisms that govern life. The central dogma of molecular biology, which supports our understanding of genetic information flow, sets the stage for the next major steps: decoding and understanding the vast amount of genetic data. Sequencing technologies have transformed our capability to analyze and interpret genetic information. The detailed analysis of genomes through these technologies has paved the way for new insights into genetic variation, evolutionary biology, and the molecular basis of diseases. In the next section, we will discuss the progression of sequencing technologies, from their early development to the advanced techniques used today, and their significant impact on biological research.

2.2 SEQUENCING

Sequencing is a process of deciphering the precise order of nucleotides in a DNA molecule, comprising several steps such as sample preparation, library preparation, sequencing, and finally, assembly and analysis. Initially, DNA is extracted from biological samples. Following extraction, the DNA is typically purified to remove contaminants. The DNA may be fragmented into smaller pieces depending on the sequencing technology. For some technologies that produce short stretches or runs of sequences, fragmentation is a standard step. However, fragmentation may not be necessary for technologies that produce long stretches or runs of sequences. Subsequently, the DNA fragments undergo library preparation, where adapter molecules are attached to facilitate the binding of the fragments to the sequencing platform, which then serves as the starting point for the DNA synthesis. The sequencing process varies based on the technologies used. The generated raw data from the sequencing step comprises stretches or runs of DNA sequences, also known as reads. The size of the reads varies between sequencing platforms. Some platforms produce short reads between 250-800 base pairs, while others produce long reads of lengths above $10,000$ base pairs [29] . These reads are then processed and analyzed to reconstruct the original DNA sequence in a process called assembly. The assembly is a post-processing step involving merging reads into contiguous sequences, optionally mapping the contiguous sequences with reference genomes, and putting all pieces together to reconstruct the original genome sequence in question. Analyzing the sequencing data reveals genetic variations, mutations, gene expression patterns, and various other genomic features.

2.2.1 GENETIC VARIATIONS

Genetic variations are differences in the DNA sequence among individuals within a population. These differences can be due to mutations, genetic recombination, and other evolutionary processes (e.g., positive selection, heterogeneous selection, gene flow, natural selection, etc., to name a few) [30]. Mutations are alterations in the sequence that can be categorized into several types. Three major classes of mutations include structural variants, indels (insertions and deletions), and point

mutations. Structural variants [31] involve large-scale alterations, such as duplications, deletions, inversions, and translocations of DNA segments, affecting multiple genes or large genomic regions . For example, duplications happen when a genome segment is copied more than once, often due to errors during DNA replication or misalignment of chromosomes during meiosis. Deletions involve the loss or removal of parts of the DNA sequence, which can result from errors during DNA repair processes. Inversions occur when a segment of DNA is flipped and reinserted in the opposite orientation, usually due to double-strand breaks that are incorrectly repaired, inverting the order of the DNA segment. Translocations usually arise from errors in chromosome segregation during cell division or from interactions between different chromosomes, leading to segments of DNA moving from one chromosome to another. These are relatively rare but can lead to significant phenotypic changes (observable traits or characteristics of an organism) or diseases [31]. Indels [32] are medium-sized genetic changes that involve the insertion or deletion of one or more bases in the DNA. Point mutations (silent, nonsense, and missense), also known as single nucleotide polymorphisms (SNPs) or single nucleotide variants (SNVs), are the most common form of genetic variation [33]. SNPs refer to point mutations occurring in at least 1% of the population, while SNVs are point mutations occurring in less than 1% of the population. The 1000 Genomes Project revealed 88 million variants, of which 84.7 million are SNPs, 3.6 million short indels, and 60,000 structural variants prevalent in the human genome [34, 35].

2.2.2 HISTORY OF SEQUENCING

In 1953, James Watson and Francis Crick revealed the double helix structure of DNA [28], emphasizing the importance of determining the exact sequence of bases. The same year, Frederick Sanger sequenced two chains of insulin protein [36], the first biological molecule to be sequenced, using a refined partition chromatography method to determine its amino acid sequence. This approach involved fragmenting the protein into smaller pieces, sequencing each fragment, and then overlapping these sequences to reconstruct the full protein sequence. Sanger's work with proteins demonstrated the feasibility of sequencing complex biological molecules and established foundational principles that would later be

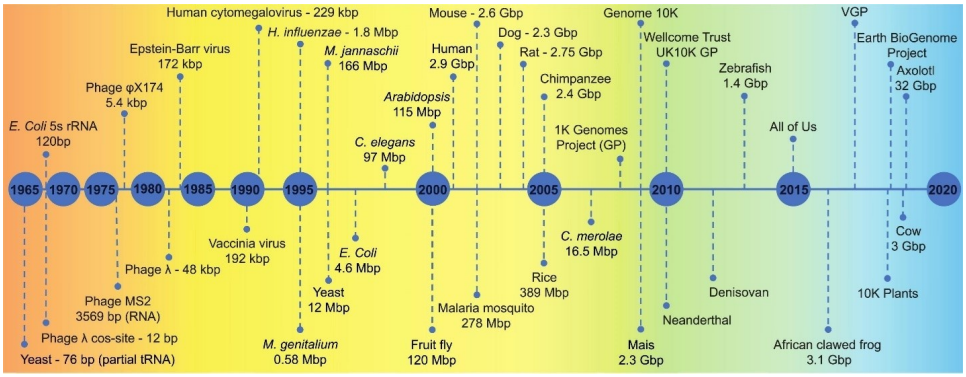

Figure 2.2: *(Source: (Giani et al., 2020 [37])) Milestones in sequencing. This timeline showcases major genome assembly achievements from the beginning of the sequencing era to current large-scale projects. Each genome or project is color-coded according to the sequencing technique used: light red for early methods, yellow for Sanger sequencing, green for next-generation sequencing, and light blue for third-generation sequencing.*

applied to nucleic acids laying the groundwork for modern sequencing [37]. Advancements in RNA sequencing started in 1965, before DNA sequencing, with the sequencing of alanine tRNA from *Saccharomyces cerevisiae* [38]. This was possible because the methods to cleave RNA fragments had been available since the 1940s [38]. Milestones in DNA sequencing included the phage λ cos-site in 1968 and the coat protein gene from phage MS2 in 1972. In 1977, Sanger sequenced the ϕX174 genome using chain-terminating ddNTPs, revolutionizing DNA sequencing [37] (Figure 2.2).

Sequencing technology has evolved significantly over the last fifty years (Figure 2.3). The first generation of DNA sequencing, pioneered by Sanger in 1977 [39], provided the foundational methodology. The second generation introduced high-throughput capabilities with platforms such as Illumina [37, 40, 41] and Ion Torrent [37, 42], enabling rapid and cost-effective sequencing of vast data compared to its previous generation. The current third generation includes technologies like PacBio [43, 44] and Oxford Nanopore [45], offering long-read and single-molecule sequencing capabilities. These advancements have significantly expanded our ability to analyze complex genomes, providing critical insights into genetic variation, evolutionary biology, and disease mechanisms, including completing the first human genome. Sequencers produce strings called reads containing the nucleotides A, T, G, C, and various IUPAC codes representing ambiguous bases [46]. Determining the order of nucleotides through sequencing technology has helped unlock the code in

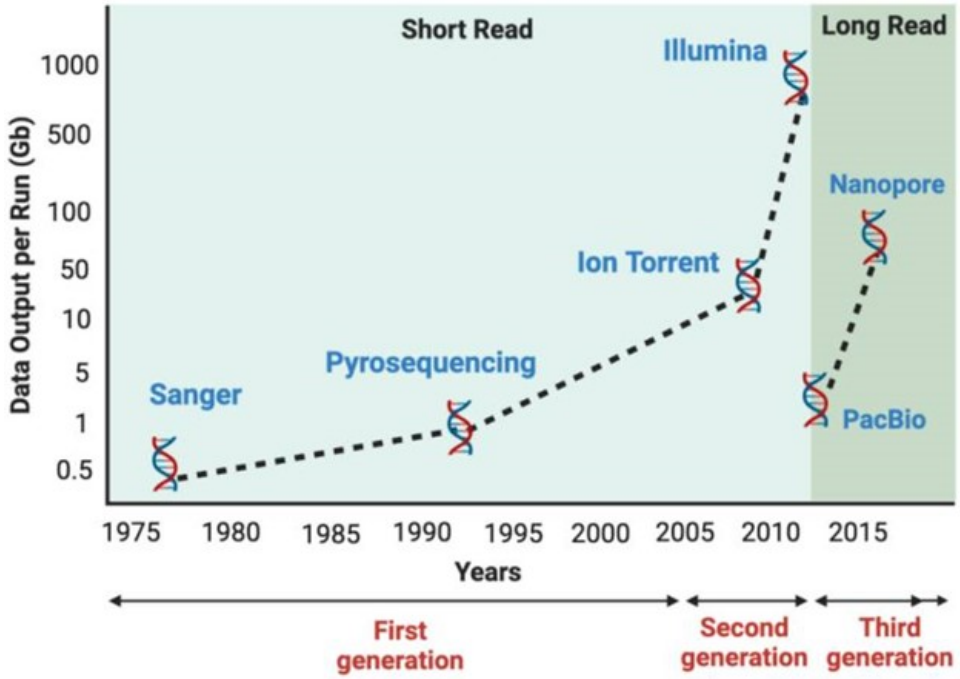

Figure 2.3: *(Source: (Satam et al., 2023 [47])) Evolution of sequencing technologies.*

several biological life forms and for understanding and treating genetic diseases.

2.2.3 FIRST-GENERATION SEQUENCING

The first generation of sequencing technologies, known as Sanger sequencing, was invented by Sanger *et al.* in 1977 [39], followed by Maxam-Gilbert sequencing, developed in the 1980s [48]. These groundbreaking techniques fundamentally established the field of DNA sequencing, enabling the study of the genetic code across various organisms and driving the subsequent development of faster and more efficient sequencing technologies used today.

Sanger sequencing uses a single DNA strand as a representative of the double-stranded DNA, and the sequencing is made by introducing chemically modified dideoxynucleotides, also known as ddNTPs (ddATP, ddTTP, ddGTP, ddCTP), during the chain elongation that leads to a termination of the elongation process. Chain elongation is a process where nucleotides are added to the growing DNA strand in the five prime (5′ end) to three prime (3′ end) direction, meaning that nucleotides are added to the 3′ end of the new DNA strand. The 5′ and 3′ refer to the

numbering of carbon atoms in the sugar moiety of the nucleotide; the 5′ end has a phosphate group attached, while the 3′ end has a hydroxyl group. During chain elongation, DNA polymerase enzymes catalyze the formation of phosphodiester bonds between adjacent nucleotides, thereby extending the DNA strand. The elongation process terminates when the DNA polymerase incorporates ddNTPs due to their lack of the 3′ hydroxyl group, which prevents the addition of further nucleotides. As a result, fragments of varying lengths are generated and then separated by gel electrophoresis. Originally, the Sanger method required the cloning of DNA fragments to generate enough for sequencing, which added complexity to the process. This method is still used today for its high accuracy, although the throughput is limited. This method was used to decipher the first ever human genome (Human Genome Project, HGP) [5], and it's assumed to have cost approximately 100 million US dollars and took nearly 15 years to complete [49].

The Maxam-Gilbert sequencing method performs sequencing without cloning , and it utilizes chemical reactions to cleave DNA at specific bases and is most effective with small nucleotide polymers [50]. This method involves labeling DNA fragments at one end and then subjecting them to chemical reactions that generate breaks at specific bases, which are later separated using gel electrophoresis. This method was less commonly used due to its complexity, as it involves using toxic and radioactive chemicals [50]; additionally, the emergence of automated Sanger sequencing, which significantly improved efficiency and throughput, made it a less attractive alternative.

2.2.4 SECOND-GENERATION SEQUENCING

Due to a lack of alternative approaches, the cost and time required for first-generation sequencing were disadvantages, even though the technology was dominant for two to three decades. Second-generation sequencing, also known as next-generation sequencing (NGS) or high-throughput sequencing (HTS) technology, emerged in the 2000s and shattered the limitations of first-generation sequencing. It has enabled massively parallel sequencing of millions of short reads from multiple samples at a much-reduced cost without the requirement for gel electrophoresis.

Short-read sequencing (Figure 2.4) is classified mainly into two approaches: sequencing by ligation (SBL) and sequencing by synthesis (SBS) [51]. SBL is a DNA sequencing technique (example: AB SOLiD sequencing platform, introduced in 2007) that involves identifying the sequence of a DNA fragment by ligating short oligonucleotide probes to the template DNA. These probes are designed to be complementary to specific regions of the DNA. Each search is labeled with a unique fluorescent tag, allowing the identification of the incorporated probe during analysis. The process begins by hybridizing the probes to the DNA template. A probe matches the template and gets ligated to the DNA strand. After ligation, the fluorescent tag attached to the probe is detected, indicating the identity of the base at that specific position. This process is then repeated for multiple probes for the sequence of the DNA fragment.

SBS utilizes DNA polymerase to synthesize a complementary strand of DNA based on the template strand (employing the polymerase chain reaction, PCR). However, each nucleotide added to the growing strand is modified with a reversible fluorescent label and a blocking group. The process involves adding one nucleotide at a time to the growing strand. When a nucleotide is added, its fluorescent label is detected, and the color indicates the incorporated base. Then, the fluorescent label and blocking group are removed, allowing the next nucleotide to be added. This repeated cycle generates a fluorescence signal sequence corresponding to the DNA sequence. The generated sequence data is processed and analyzed using both techniques to reconstruct the original DNA sequence.

The emergence of these NGS techniques played a pivotal role in genomics, personalized medicine, metagenomics, epigenomics, and several other OMICS research. For instance, 2003 marked the development of Solexa's SBS method, and Solexa was later acquired by Illumina in 2007 [37, 40, 41]. Illumina's advancements include high throughput and scalability, enabling rapid and cost-effective whole-genome sequencing. However, its relatively short read lengths are a limitation [52]. Roche's 454 [53] sequencing method, launched in 2010, determines DNA sequences through pyrosequencing by detecting pyrophosphate release and light generation during nucleotide incorporation. This method provides read lengths of 400-500 base pairs and relatively high accuracy.

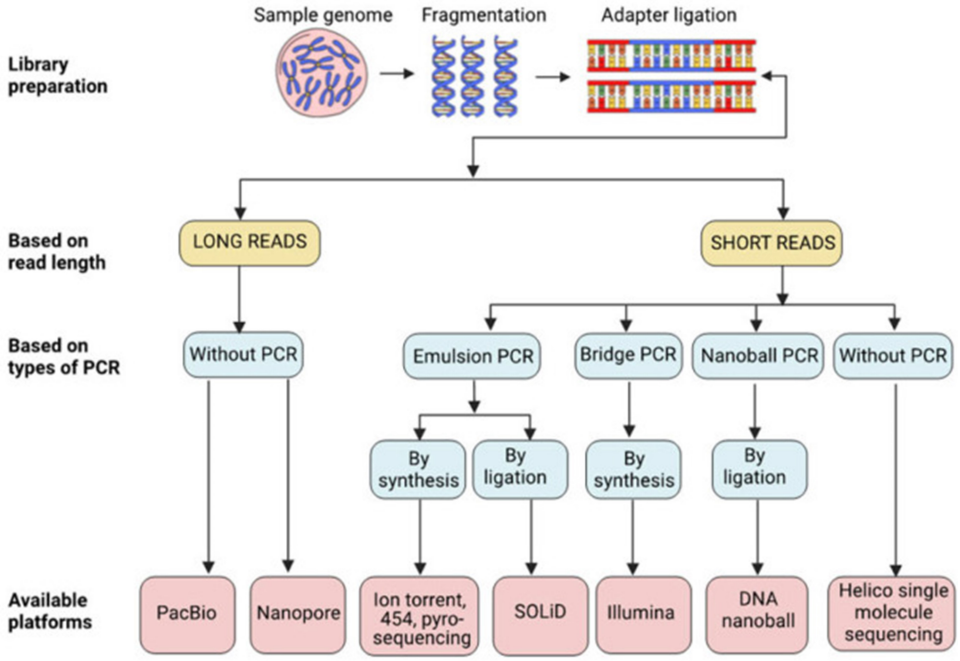

Figure 2.4: *(Source: (Satam et al., 2023 [47])) Overview of sequencing technologies of different platforms that use different methods for sequencing.*

It can sequence up to one billion bases a day but with limitations in cost and accuracy in homopolymeric regions [37, 41, 47]. Ion Torrent [42], launched in 2010, uses semiconductor sequencing technology. This approach measures the pH variations induced by the release of hydrogen ions during DNA synthesis to determine the DNA sequence. This method allows for faster and more cost-effective sequencing, though it also faces challenges with homopolymer regions and shorter read lengths [37, 41, 47, 54].

These advancements have significantly accelerated sequencing speed and increased capacity, making it possible to conduct whole-genome sequencing, transcriptome analysis, and targeted sequencing more efficiently than ever before. The major NGS platforms were Roche/454, Illumina/Solexa, and ABI/SOLiD until the late 2010s [50]. This leap has been crucial for exploring genetic variation, understanding disease genomics, and advancing personalized medicine. For example, Roche's 454 sequencing technology took only two months to sequence the human genome and cost approximately one-hundredth of the cost compared to the first-generation sequencing [49]. In contrast, Ion Torrent claimed that their Ion Proton sequencer could be used to sequence the human genome in a day and for the price of a thousand dollars [37]. However,

due to their limitations, these platforms have declined in usage, and the SBS method, taken over by Illumina, ultimately became the dominant approach and the de facto choice for a wide range of genomic applications in the 2010s, accounting for more than 90% of all DNA sequence data produced [37].

2.2.5 THIRD-GENERATION SEQUENCING

Third-generation sequencing is a cutting-edge DNA sequencing technology that has exceeded the limits of previous generations, which emerged at the end of the 2010s. It offers innovative solutions to the challenges of prior methods and represents the latest technological developments. Unlike the earlier generations, it offers long read lengths, real-time data collection, and direct sequencing. This contributes to increased accuracy, efficiency, and capacity for various applications.

A prominent example of third-generation sequencing is the single-molecule, real-time (SMRT) sequencing of Pacific Biosciences (PacBio) [41, 43, 44]. In SMRT sequencing, a DNA polymerase enzyme and a single DNA molecule (template) are immobilized on a small, transparent well, allowing for the observations of DNA synthesis in real-time as nucleotides are incorporated. This process produces long reads that span thousands to tens of thousands of nucleotides [42, 55]. Another third-generation sequencing method includes nanopore (a nanopore is a hole of nanometer size formed by proteins or synthetic materials) sequencing from Oxford Nanopore Technologies [45]. This method involves passing a single DNA strand through nanopores embedded in a membrane. As individual nucleotides pass through the nanopore, they cause distinct changes in electrical current. These electrical changes are detected and recorded in real-time, generating a unique electrical signal for each nucleotide, enabling the determination of the DNA sequence [37, 41].

Both first and second-generation technologies require the PCR amplification step (Figure 2.4), which often increases cost and time [47, 56]. Also, genomes are very complex and have many long, repetitive elements, copy numbers, and structural variations [47, 54, 57, 58]. However, many complex elements are long, and short-read technologies cannot solve them, making genome assembly more difficult. Third-generation sequencing solves several problems of previous generations but at the

cost of a higher error rate. However, error correction strategies and improvements in data analysis have contributed to the practical utility of third-generation sequencing in OMICS research [41, 47, 55].

2.3 SEQUENCE DATA FORMATS

Sequence data is managed using specialized file formats, primarily FASTA [59] and FASTQ [60]. The FASTA format encodes nucleotide and protein sequences, whereas the FASTQ format, commonly used for NGS reads, includes both nucleotide sequences and corresponding quality scores. FASTA is a widely used text-based format for representing nucleotide or protein sequences. Each entry in a FASTA file begins with a single-line description starting with a greater-than (">") symbol, followed by the sequence identifier and an optional description. The subsequent lines contain the sequence data, a continuous string of letters for nucleotides or amino acids. Meanwhile, the FASTQ format extends the FASTA format by including quality scores for nucleotide sequences, which are essential for evaluating the accuracy of high-throughput sequencing reads. Each FASTQ entry consists of four lines: the first line starts with an "@" symbol, followed by the sequence identifier; the second line contains the nucleotide sequence; the third line begins with a "+" symbol, optionally followed by the same identifier as the first line; and the fourth line contains the quality scores. The quality scores are ASCII characters derived from the Phred quality scores [61, 62]. The Phred scores indicate the probability of an error in base calling, the process of converting raw sequencing data from high-throughput sequencing machines into nucleotide sequences. Higher scores reflect greater confidence in the accuracy of each nucleotide.

2.4 SEQUENCE ANALYSIS

Sequence analysis is the process of understanding and interpreting the genetic information encoded within DNA, RNA, or protein sequences. With the advent of NGS or HTS technologies, this field has grown immensely due to the overwhelming amount of sequencing data generated and the reduced costs of sequencing experiments [63]. This presents unique opportunities and significant challenges in sequence similar-

ity search. The crucial methods for sequence similarity search are categorized into alignment-based (reference-based) and alignment-free (reference-free) approaches.

2.4.1 ALIGNMENT-BASED APPROACH

Alignment-based sequence analysis is the central approach in genomics that involves comparing sequenced reads to a known reference genome, allowing for the identification of similarities and differences [64]. Reference genomes are high-quality, well-annotated, and assembled sequences of DNA that serve as a representative template for a particular organism or a species [65]. The methods of this class offer insights into genetic variations and functional elements by aligning the sequenced reads to a reference. Sequence analysis through this method encompasses the following steps: read mapping with pre- and post-processing stages, variant calling and annotation, and genotype determination [66].

Reference-based analysis helps to identify disease-associated mutations and population-specific variations to compare genomes across species, infer evolutionary relationships, and identify conserved regions [66]. The advantage of such methods is the accurate identification of known genetic variations, which facilitates the interpretation of the functional impact of variants based on the available annotations. However, this method leads to a well-known problem, reference bias; it does not correctly represent the population, leading to the underrepresentation of specific alleles or genetic variants [67]. It might overlook variants not present in the reference genome and falsely identify or misrepresent variants that are incorrectly included, particularly in less-studied populations.

Several alignment-based methods have been proposed in the last few decades [64, 68–71]. Algorithms like Needleman-Wunsch [72] and Smith-Waterman [73] use dynamic programming to find the optimal global and local alignments given two sequences. Aligning two sequences is referred to as pairwise sequence alignment. This key method compares and aligns two biological sequences (proteins or DNA) to determine their similarities and differences. The aim is to align the sequences to match the maximum number of identical or similar characters, allowing for gaps to be introduced where necessary to optimize this alignment.

Sequence identity refers to the proportion of identical characters between two aligned sequences, providing a measure of their similarity.

The larger set of possible characters with distinct chemical properties increases the complexity of protein sequence alignment. Therefore, protein sequence alignment methods use substitution matrices, such as BLOck SUbstitution Matrix (BLOSUM) [74] or Point Accepted Mutation (PAM) matrix [75], to assign scores for matches, mismatches, and gaps based on their similarity. These matrices help align sequences, even when sequence identity is low, by assigning positive scores to similar amino acids and penalties to dissimilar ones.

Local alignment identifies regions of high similarity within subsequences, making it ideal for only partially overlapping sequences. In contrast, global alignment aims to align entire sequences end-to-end, which is best suited for comparing two closely related sequences. Following the optimal alignment methods, heuristic algorithm-based tools like BLAST were developed in 1990 and are used to find regions of local similarity between sequences. It starts by finding short exact matches (seeds) between the query and database sequences. These seeds are then extended in both directions to form High-scoring Segment Pairs (HSP). BLAST can use a scoring matrix (e.g., BLOSUM and PAM) to assign scores to matches, mismatches, and gaps. The statistical significance of alignments is evaluated using E-values (Expect value), which measure the number of times one can expect to find a given sequence match by chance in a database of a particular size, where lower E-values indicate significant matches.

In later years, due to the emergence of NGS, sophisticated algorithms and tools were required to efficiently align or map millions or billions of genomic reads to reference genomes. While effective for smaller datasets, traditional alignment tools are overwhelmed by NGS's massive volume of data. Tools like BLAST were designed for queries against relatively small databases, leading to significant bottlenecks when processing millions or billions of reads from NGS experiments. Additionally, the computational requirements for BLAST increased exponentially with the size of the dataset, making it impractical for large-scale genomic studies. This inefficiency emphasized the need for sophisticated algorithms to handle high-throughput sequencing data with greater speed, scalability, and accuracy. Burrows-Wheeler Aligner (BWA) [76] and Bowtie [77] are

two of the most prominent tools, among others. BWA is based on the Burrows-Wheeler Transform (BWT) [78] method that rearranges the text into a more compressible format by sorting its rotations lexicographically. This process results in strings with runs of identical characters, thereby significantly increasing the compression ratio and query efficiency [76]. The tool supports gapped and ungapped alignments and can handle varying read lengths, from 70 base pairs to several kilobases. Bowtie is another essential tool that is noted for its speed and minimal memory usage. Bowtie employs the BWT and FM-index [79] to map millions of reads efficiently. Bowtie uses a backtracking algorithm to align reads, enabling it to quickly find optimal alignments for short reads (better performance for less than 50 base pairs) [77]. While Bowtie does not allow gaps in alignments, Bowtie2 [80], an improved version, supports gapped alignments to handle indels. BWA is optimized for sensitivity and accuracy, while Bowtie focuses on speed and efficiency [64, 68–70].

2.4.2 ALIGNMENT-FREE APPROACH

Alignment-free methods provide an alternative approach to traditional alignment-based methods. In the case of genome assembly, instead of aligning or mapping reads to a reference, these methods assemble the sequenced reads into longer contiguous sequences, called contigs, without relying on a template by piecing together the overlapping segments from the reads, a process called *de novo* assembly [81]. Since references significantly influence the quality of the downstream analysis, some experiments cannot rely on them. In the case of metagenomes, for example, a reference-based alignment does not apply to the full sample because the dataset may comprise numerous organisms for which the reference genome may not exist [82, 83]. This is especially true for populations yet to be thoroughly explored. A wide range of alignment-free approaches to sequence comparison have been developed. These approaches include methods based on graph algorithms, k-mer (subsequences of length k, also referred to as words in this context) or word counts, chaos theory, information theory, and distance-based methods, among others [84]. In the following, I briefly present a couple of these methods.

The graph-based methods [85] are divided into the Overlap-Layout-Consensus (OLC) [86] and De Bruijn Graph (DBG) [64] methods. The

OLC method is a classical approach that involves three key steps: overlap, layout, and consensus. Assemblers start by detecting the overlaps among the reads; in the layout step, these overlaps are used to construct a graph where nodes represent reads and edges represent overlaps, and the aim is to find the shortest Hamiltonian path that visits each node in the overlap graph exactly once thereby representing an optimal overlap of reads. Finally, the identified overlaps are combined in the consensus step to create a final sequence. Some of the tools based on OLC include PHRAP [87], CAP3 [88], and Celera [89]. Along with OLC , the DBG-based method is another widely used approach. DBGs are constructed by decomposing the reads into overlapping *k*-mers, which are then used as nodes in the graph, and the edges are created between nodes that overlap by *k*-1-mers. This allows for the reconstruction of sequences by traversing the graph, making DBGs effective in handling repetitive regions and short reads. At the same time, OLC is more accurate with long reads but less practical for high-coverage short reads due to its high computational requirements. Euler [90], Velvet [91], ABySS [92], and SPAdes [93] are some of the tools based on DBGs [85].

The *k*-mer or word count-based methods [63, 82–84, 94, 95] is also prominent in genomic sequence analysis, which uses the frequencies of *k*-mers that can be used to identify patterns, classify sequences, detect genomic similarities and differences across datasets, annotate functional regions, identify variants, and sequencing error correction. The approach involves several key steps: counting *k*-mers, calculating similarity/dissimilarity measures, and clustering sequences. In the first step, tools like Jellyfish [96], DSK (Disk Streaming of *k*-mers) [97], and KMC [98] are used to count occurrences of *k*-mers in a sequence. The second step involves computing similarity or dissimilarity between sequences based on *k*-mer frequencies. Different metrics, such as Euclidean distance, Manhattan distance, Jensen-Shannon divergence, Hamming distance, and Jaccard index, can be used [94, 95]. Some methods, like feature frequency profiles (FFP), rely on observed word frequencies, while others, such as d_2, d_2^*, and CVTree, incorporate background word frequencies to account for expected *k*-mer occurrences and enhance accuracy [99–104]. In the last step, clustering algorithms such as hierarchical clustering and neighbor-joining use these similarity measures to group sequences together. The *k*-mer approach is computationally efficient and scalable,

making it well-suited for large-scale genomic datasets, metagenomics, and *de novo* sequence analysis [94].

2.5 CHALLENGES IN SEQUENCE SEARCH: THE CASE FOR METAPROFI

The explosion of genomic information, driven mainly by NGS technologies, has reached unprecedented amounts of biological sequencing data available in public databases, as is evident from examples of the European Nucleotide Archive (ENA) [6] or Sequence Read Archive (SRA) [7]. As of February 2024, SRA contains (Figure 2.5) approximately 91 petabases, including 53 petabases that are open access, meaning they are freely available to the public without restrictions. These massive amounts of data pose significant computational challenges for bioinformatics analysis, especially in sequence similarity search. Although traditional alignment-based methods such as BLAST work well for smaller datasets, they face significant challenges when dealing with vast volumes of today's genomic data, especially in analyzing metagenomic samples containing diverse and complex mixtures of organisms. Alignment-based approaches inherently depend on reference genomes, which may not exist for many species, resulting in biases and potential inaccuracies. This dependency is a significant limitation, especially for analyzing novel or less-studied organisms with sparse references. While alignment-free methods offer an alternative by eliminating the need for reference sequences, they, too, encounter challenges due to the ever-increasing size of genomic datasets.

These databases offer significant potential for scientific advancements but also introduce major challenges regarding storage, efficient access, and analysis. As the size of the databases reached the petabyte scale, it has become difficult to support online searches in these databases. This rapid growth creates a significant demand for effective management, and the need for tools to process, store, and query these extensive collections of sequence data without high memory and storage requirements constitutes a significant computational challenge. We address these issues through MetaProFi (Chapter 4), our novel tool that offers a scalable and efficient solution for sequence similarity search by integrating the advantages of alignment-free approaches with advanced

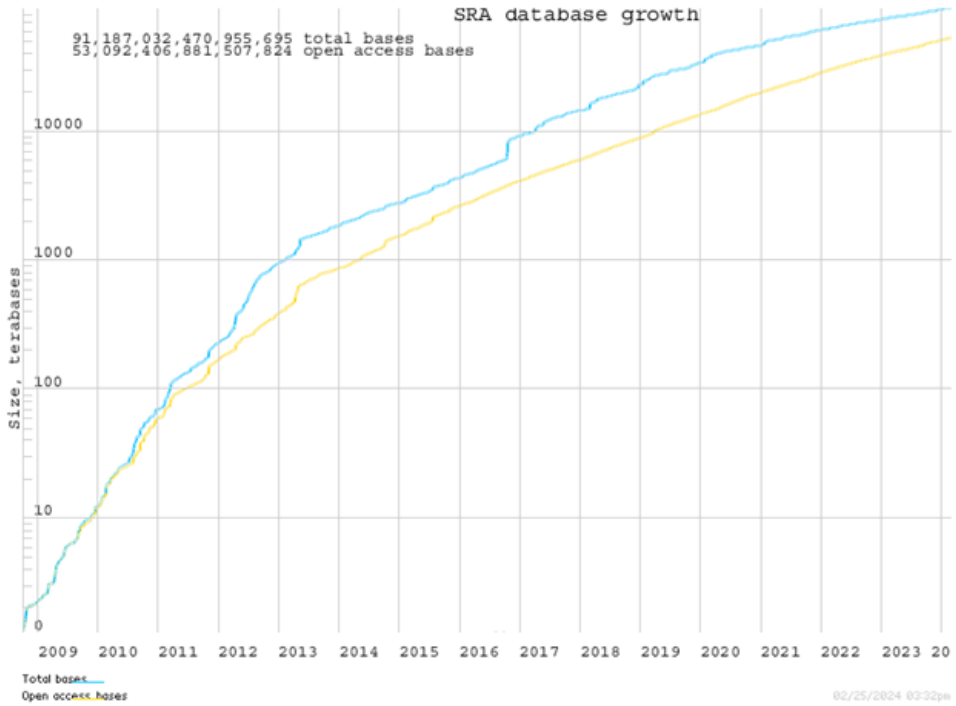

Figure 2.5: *(Source: https://www.ncbi.nlm.nih.gov/sra/docs/sragrowth/) Sequence Read Archive database growth over the time.*

computational techniques. MetaProFi's ability to efficiently index and query nucleotide and protein sequences sets it apart from the existing methods. This capability is crucial given the increasing complexity and size of sequence databases. MetaProFi ensures real-time analysis and accurate identification of functionally relevant genetic variants, making it an important tool for modern bioinformatics.

2.6 PROTEINS: SEQUENCE TO FUNCTION

Proteins are essential molecules crucial in every biological process within living organisms. They have diverse functions, including enzymatic catalysis of biochemical reactions, providing structural support to cells and tissues, regulating gene expression, facilitating cell signaling and cell cycle, mediating immune responses, and transporting molecules. Understanding proteins, their structure, and their function is essential to analyzing and interpreting biological data, as they are central to elucidating disease mechanisms, understanding drug resistance, tracing evolutionary relationships, and developing targeted therapies in precision medicine. Protein synthesis can be viewed as a part of the central

dogma of molecular biology [4], which describes the flow of genetic information within a biological system. It states that genetic information is transferred from DNA to RNA through a process called transcription and from RNA to protein through a process known as translation. Transcription is the process of transcribing DNA into the mRNA in the cell nucleus. The mRNA serves as a template that carries the genetic code from DNA to the ribosome, where protein synthesis occurs. During translation, ribosomes read the mRNA sequence and translate it into a specific sequence of amino acids, the building blocks of proteins. This sequence of amino acids ultimately folds into a functional protein.

2.6.1 AMINO ACIDS: THE BUILDING BLOCKS

Amino acids [105] are organic molecules that serve as the building blocks of proteins. There are 20 naturally occurring amino acids, each with a unique side chain that determines their chemical properties and role in protein structure. Each amino acid (Figure 2.6) consists of a central carbon atom (α carbon) bonded to four different groups: an amino group (NH_2), a carboxyl group (COOH), a hydrogen atom (H), and a variable side chain (R group) that differs among amino acids and determines the amino acid's identity and properties. The diversity of side chains allows amino acids to form proteins with many structures and functions. These side chains can be polar, nonpolar, charged, or aromatic; they are how proteins fold and interact with other molecules. Peptide bonds link amino acids together to form proteins. A peptide bond (Figure 2.7) is a covalent bond that forms between the carboxyl group of one amino acid and the amino group of another, releasing a water molecule in the process. When more amino acids are added, this becomes a polypeptide, which eventually folds into a functional protein.

The rigidity of the peptide bond restricts the polypeptide chain's flexibility, allowing rotation only around the bonds connected to the alpha carbon (Cα). Two angles capture these rotations: Phi (ϕ), which is the angle around the N-Cα bond, and Psi (ψ), which is the angle around the Cα-C bond. These angles are limited by the steric hindrance or spatial clashes between atoms, ensuring that the protein structure avoids unfavorable overlaps and maintains stability. The Ramachandran plot [106] is used to visualize the permissible combinations of these

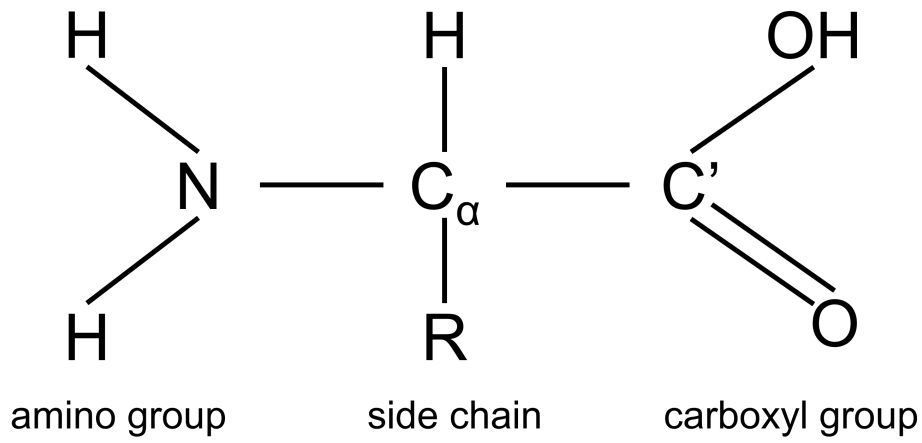

Figure 2.6: *Structure of an amino acid showing the alpha (α) carbon, an amino group, a carboxyl group, a hydrogen atom, and the variable side chain (R group).*

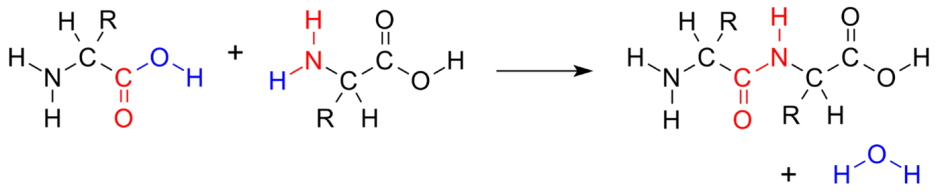

Figure 2.7: *(Source: https://en.wikipedia.org/wiki/Peptide_bond) A peptide bond (red atoms) forms between two amino acids, leading to the release of water and the creation of a dipeptide.*

angles, aiding in predicting and analyzing the three-dimensional (3D) folding patterns of proteins.

2.6.2 PROTEIN FOLDING AND HIERARCHY

Protein folding is the process by which a linear chain of amino acids acquires its 3D structure, and studying and understanding protein structure is crucial to understanding its biological function [107]. Molecular chaperones are a group of proteins that assist in folding, ensuring proteins fold into their native conformation [108]. Protein structure (Figure 2.8) is classified into four levels: primary, secondary, tertiary, and quaternary structures [109]. Each contributes to the protein's overall shape and function.

The protein's primary structure, which is its linear sequence of amino acids, fundamentally determines its higher-order structures, as demonstrated by Christian Anfinsen in 1973 [9]. However, environmental factors, such as interactions with ligands, substrates, or other proteins, can influence the 3D structure, leading to conformational changes. Additionally, some proteins contain intrinsically unstructured or disordered regions [110] that do not adopt a fixed structure under physiological conditions.

Secondary structure refers to the local spatial arrangement of a protein's polypeptide chain, independent of its overall conformation. This is due to the specific folding patterns and constraints imposed by peptide bonds and hydrogen bonding interactions between the backbone atoms of the polypeptide chain. The two prominent types of secondary structure elements are the alpha (α) helix and beta (β) sheet, based on the physical constraints of polypeptide chains [109, 111, 112]. These structures are stabilized by hydrogen bonds between backbone atoms, making them energetically favorable [113]. While α helices and β sheets are the primary regular secondary structure elements, proteins also contain other secondary structure elements, such as loops or coils and β turns. Often found on the protein's surface, these loops along with other secondary structure elements can serve functional roles such as forming active sites [113]. The active site is a specific region, typically a pocket on the protein's surface, where substrate molecules bind and undergo a chemical reaction.

The tertiary structure represents the overall 3D shape of a single polypeptide chain formed by the interactions between secondary structure elements and amino acid side chains [113]. It is the functional biological structure of most proteins. The overall shape of the tertiary structure of a protein is commonly referred to as a fold, which describes how secondary structure elements, like α helices and β sheets, are arranged in a 3D space. Folds can be associated with specific functions, such as binding to nucleic acids, small molecules, or other proteins, as well as facilitating enzymatic catalysis, stabilizing protein structures, and mediating signal transduction pathways [109, 114, 115] . Folds often comprise distinct structural units known as domains and motifs, contributing to their overall function and evolutionary history [109]. Domains are compact, structurally stable regions within a protein, typically consisting of 50 to 200 amino acids. They are identified based on their sequence similarity and ability to fold into a stable, 3D structure independently [116]. Larger proteins often comprise multiple domains, each with unique functions or contributing to the protein's overall structure [117]. While the term domain is sometimes used interchangeably with fold and motif, domains are generally defined by conserved sequences and functional behavior, making them fundamental protein structure and function units [109, 114, 118].

A structural motif is a smaller, repetitive unit within a protein that forms part of these larger folds [119]. Unlike the broader folds, these motifs are generally smaller and can be crucial in defining the protein's function. They often play critical roles in stabilizing protein structures and facilitating the interaction with other molecules. Common examples of structural motifs include β-barrels and zinc fingers, which are frequently observed motifs. β-barrels consist of β strands arranged in a cylindrical shape, forming a barrel-like structure. It is often seen in transport proteins and enzymes, where it aids in forming channels or binding sites. The zinc finger is a highly conserved motif stabilized by a zinc ion coordinated with cysteine and histidine residues, forming a finger-like structure. It typically includes an α helix and β strands and is crucial for DNA binding and gene regulation [109, 120]. While motifs are smaller and can be repeated across different proteins, domains are larger and can encompass several motifs, contributing to the overall protein architecture and function. Many proteins function not as a single polypeptide chain

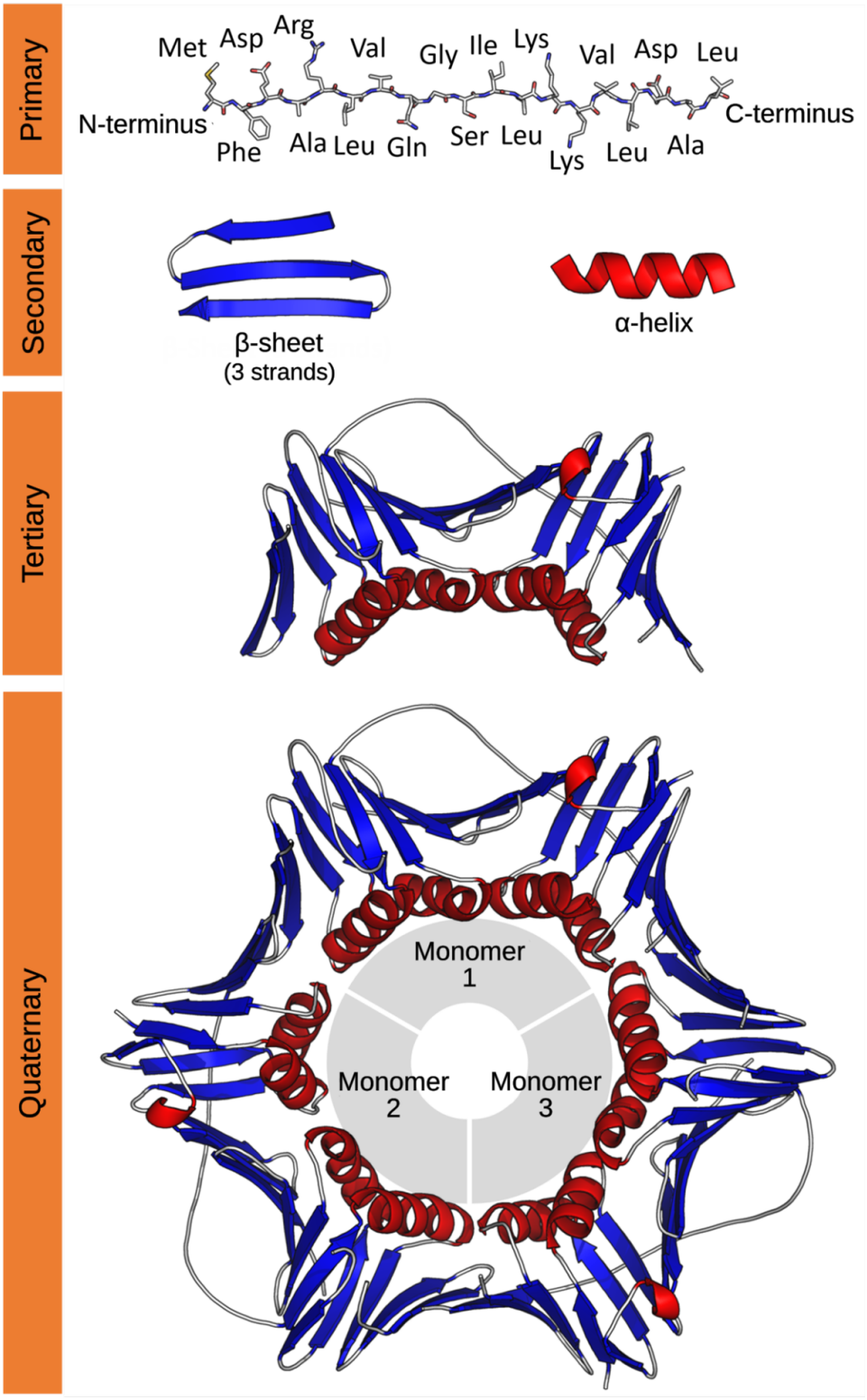

Figure 2.8: *(Source: https://commons.wikimedia.org/wiki/File:Protein_structure_(full)-en.svg by Jmarchn, from Thomas Shafee. Licensed under CC BY-SA 3.0 via Wikimedia Commons) Overview of protein structure hierarchy on the example protein PCNA (PDB ID: 1AXC). The figure highlights the four levels of protein structure: primary, secondary, tertiary, and quaternary.*

(subunit) or monomer but as a complex of multisubunits or multimers. This structure refers to the precise spatial arrangement and interaction of two or more independently folded polypeptide chains into a single functional protein called the quaternary structure. These subunits can be identical, forming a homomeric protein, or different, resulting in a heteromeric protein. Various non-covalent forces stabilize their interactions, such as hydrogen bonds, ionic interactions, and hydrophobic effects [121].

Studying and understanding the protein sequence-to-function relationship requires thoroughly understanding their 3D structures. As previously discussed, folding chains of polypeptides into specific conformations defines their functional roles in biological processes. However, elucidating the precise structures of proteins poses significant challenges due to the complexity and dynamic nature of protein folding [122]. Experimental and computational methods have been developed to tackle these challenges, providing insights into molecular architecture. These techniques allow us to visualize the spatial arrangement of atoms within a protein, facilitating studying protein interactions, dynamics, and functions. The ability to accurately determine protein structures is essential for advancing our understanding of biological processes and developing new therapeutics. Complete structural information reveals how proteins function, their dynamics, their interaction with other molecules, and their contribution to cellular processes. This knowledge is vital for designing targeted drugs and understanding disease mechanisms at a molecular level. Therefore, experimental and computational methods to determine protein structures have become key tools.

2.7 EXPERIMENTAL METHODS TO DETERMINE PROTEIN 3D STRUCTURES

The first 3D structure of a protein (myoglobin) was experimentally resolved in 1958 [123], marked a pivotal advancement in structural biology, revealing the first detailed view of a protein's atomic arrangement through X-ray crystallography [124]. X-ray crystallography [125] is a crucial method for determining the 3D structures of proteins. The process begins with protein crystallization, where the protein must form a crystal lattice. This often involves a lengthy trial-and-error procedure,

as finding the right conditions for crystallization can take years [124]. Once a suitable crystal is obtained, it is subjected to X-ray diffraction. X-rays pass through the crystal, and they are scattered by the electrons in the protein, creating a diffraction pattern. Analyzing this pattern allows for constructing the protein's electron density map, from which the 3D structure can be determined. The advantage of X-ray crystallography is its ability to provide high-resolution structural details of proteins. However, it has limitations, including difficulty crystallizing membrane proteins and large complexes.

Nuclear magnetic resonance (NMR) spectroscopy [126] is another method to determine 3D protein structures at the atomic level, which is particularly useful for proteins that are challenging to crystallize. The first step in NMR spectroscopy is to dissolve the purified protein in a buffer solution, ensuring it remains stable. The solution is then placed in an NMR spectrometer with a strong magnetic field. Radiofrequency waves are then used to perturb the nuclei of the atoms in the protein. NMR measures the interactions between atomic nuclei, providing information about their distances and angles. This data reconstructs the protein's 3D structure, revealing its conformation and dynamics. NMR is advantageous because it can be applied to proteins in their native, solution-state environments, providing insights into protein structure dynamics and multiple conformations. This makes NMR invaluable for understanding how proteins behave in physiological conditions and capturing transient states that other methods might not capture. However, NMR has limitations, including lower resolution than X-ray crystallography and the size of the protein.

The third major experimental method used to determine the 3D structure of proteins and large molecular complexes is cryo-electron microscopy (cryo-EM) [127, 128]. The process involves rapidly freezing protein samples to around -180 °C using liquid ethane to preserve them in a near-native state and then imaging them using an electron microscope. Unlike traditional electron microscopy, Cryo-EM does not require staining or embedding, which helps maintain the protein's structural integrity [128]. The frozen samples are exposed to an electron beam, and the resulting images are collected from various angles. These images are then reconstructed using computational algorithms to generate a 3D protein structure. Cryo-EM is beneficial as it can handle large, complex

proteins and assemblies that are difficult to crystallize. It also allows for observing proteins in a state that closely resembles their native state. However, Cryo-EM has limitations, including lower resolution than X-ray crystallography and challenges related to image processing for really complex and large macromolecules [127].

2.8 COMPUTATIONAL METHODS TO DETERMINE PROTEIN 3D STRUCTURES

Computational methods play a crucial role in determining and aiding in the understanding of the protein structures in this rapidly advancing field [129, 130]. With the advent of NGS, the availability of sequence data has exploded. This surge in data, coupled with advancements in analytical tools, has become a major driving force in developing and applying computational methods to determine protein 3D structures and bridge the protein sequence to structure gap [129]. These methods complement the experimental techniques by offering insights into protein folding, dynamics, and interactions at an atomic level in a time and cost-effective manner. Computational methods such as homology modeling, *ab initio* modeling, molecular dynamics (MD) simulations, and protein threading have become indispensable tools. In recent years, the widespread development and application of deep neural networks and advancements in artificial intelligence (AI) and computational algorithms have led to significant breakthroughs in predicting protein 3D structures through tools like AlphaFold [131]. Computational methods enable the exploration of protein structures when experimental data is limited or unavailable, allowing for the hypothesis of the 3D structures of proteins, predicting functional sites, and understanding the molecular mechanisms. We will present the most widely used methods, such as homology modeling, *ab initio* modeling, and molecular dynamics, that are essential for understanding and predicting protein structures and their dynamics. Methods like molecular dynamics are particularly relevant to the work discussed in this thesis.

Homology modeling [113, 132], also known as comparative modeling, is used to predict the 3D structure of a protein using a known structure of a related homologous protein as a template. This method is based on the principle that proteins with similar sequences often have similar

structures. Protein structures are more stable and evolve more slowly than the associated sequences. As a result, even when sequences undergo changes, the overall structure remains largely conserved. Thus, proteins with similar amino acid sequences typically adopt almost identical structures, and even proteins with distantly related sequences can fold into similar structures [113, 133, 134]. Homology modeling begins by searching structural databases (Chapter 2.9), such as the Protein Data Bank (PDB) [135], for homologous sequences with known structures. The target protein sequence is compared to these sequences through sequence alignment approach (Chapter 2.4.1). Once a suitable homologous protein with a known structure is identified, this structure is used as a template to build, optimize, and validate a structural model of the target protein's 3D structure. Homology modeling is highly effective when reliable templates with significant sequence identity are available and can provide insights into the function and interactions of proteins. SWISS-MODEL [136] and MODELLER [137, 138], among several others, are the most commonly used tools.

Ab initio modeling [113, 139, 140], or template-free modeling, predicts protein structures from amino acid sequences without relying on existing templates from homologous proteins. This method is based on Christian Anfinsen's principle that a protein will naturally fold into its most stable and energetically favorable state, determined solely by its amino acid sequence [9, 113]. An energetically favorable conformation is one that minimizes the protein's free energy [129]. The prediction process involves exploring different possible structures of the target protein by using libraries of known structural fragments derived from experimentally resolved structures of short sequences. Techniques such as molecular dynamics or Monte Carlo simulations [141] are employed to explore these conformations and identify those with the lowest energy, reflecting a stable structure [140]. This low-energy conformation is then refined to resemble the native structure of the protein closely. I-TASSER [142] and Rosetta [143] are the most commonly used tools for this type of modeling, enabling the study of proteins with unknown templates.

Molecular dynamics (MD) [144–146] is a computational method that simulates the movements of atoms and molecules over time, providing insights into the dynamic behavior of proteins and other biomolecules. The simulation begins with the system preparation step, where the

protein's 3D structure is set up, typically derived from experimental and/or computational methods. The protein is placed in a virtual box, typically surrounded by water molecules and ions to mimic physiological conditions. Next, energy minimization is performed to resolve steric clashes and ensure the structure is in an energetically favorable conformation, corresponding to local minima on its potential energy surface. The system then undergoes equilibration, starting with an NVT (constant Number of particles, Volume, and Temperature) step. During this phase, the number of particles, the volume of the simulation box, and the temperature are held constant while the system's pressure is allowed to vary to stabilize the system's density. This is followed by an NPT (constant Number of particles, Pressure, and Temperature) step, where the number of particles, pressure, and temperature are kept constant while the volume of the simulation box is adjusted. The NPT step ensures the system reaches the correct density and overall stability before moving to the production phase.The simulation uses a force field to model the interactions between atoms. Finally, during the production phase, Newton's laws of motion are applied to track the trajectory of each atom, providing insights into protein folding, interactions, and responses to external forces.

MD is essential in studying protein dynamics, offering insights beyond the static conformations provided by experimental methods. By simulating protein movements at the atomistic level, MD reveals how proteins undergo conformational changes when interacting with ligands and other biomolecules, which is essential for understanding complex biological processes and developing improved therapeutic strategies. These simulations are vital for exploring protein-ligand binding interactions, a fundamental aspect of drug design, allowing for the identification of key binding sites. MD simulations also provide insights into how mutations affect protein stability and functionality, contributing to our understanding of disease mechanisms at the molecular level. This is particularly important in studying diseases like cancer, where protein misfolding and altered dynamics play crucial roles in tumorigenesis and resistance [147, 148]. This method enhances our understanding of molecular biology and is crucial in developing targeted therapeutics. Hence, we have leveraged MD simulations in this thesis (Chapter 3) to study the impact of specific mutations on protein conformations and

dynamics, providing a deeper understanding of their role in disease mechanisms.

2.9 PROTEIN SEQUENCE AND STRUCTURE DATA SOURCES

Two primary repositories for protein data are the Universal Protein Knowledgebase (UniProtKB) [149] and the Protein Data Bank (PDB) [135] database, which includes comprehensive information about protein sequences and their structures. The PDB is a crucial resource that archives 3D structures of proteins and other molecules determined through experimental methods discussed earlier. PDB assigns a unique four-character code that identifies each entry in the PDB, and the 3D data is organized in the PDB file format, a standard text format used to describe 3D structures of proteins, nucleic acids, and other macromolecules. Each PDB file consists of a series of records, each providing specific information about the structure. The ATOM record lists the coordinates and other details of atoms in the protein, including atom name, residue name, chain identifier, and atomic coordinates (x, y, z). HETATM records are similar but used for non-standard residues or ligands.

On the other hand, UniProtKB is a database that offers detailed protein sequences and functional information. It is divided into two main sections [150]: Swiss-Prot and TrEMBL. Swiss-Prot contains manually curated, reviewed sequences with extensive annotations on protein function, sequence features, and biological roles. With the surge in genome sequencing, the amount of sequence data has significantly increased, leading to the development of TrEMBL. TrEMBL entries include computationally predicted and unreviewed sequences. The sequences are stored in the FASTA format, among other formats.

3

MOLECULAR DYNAMICS INSIGHTS INTO MUTATION-INDUCED STRUCTURAL CHANGES AND DRUG RESISTANCE: CASE STUDIES OF THE RECEPTOR PROTEIN TYROSINE KINASE KIT AND THE NS_3 PROTEASE FROM HCV

In this chapter, I present two molecular dynamics (MD) studies based on my published work that explore the impact of mutations associated with drug resistance. The chapter is structured as follows: it begins with a unified materials and methods section that details the methodologies employed across both studies. Following this, each sub-chapter begins with a detailed background and literature review of the proteins studied, followed by the results and discussion specific to each study, and concludes with a summary of findings.

The first case-study (Chapter 3.2) is based on the publication titled "A shift of dynamic equilibrium between the KIT active and inactive states causes drug resistance" [10], of which I was the first author. My contribution to this work involved designing and performing all the molecular dynamics (MD) simulations, analyzing the resulting data, and writing the manuscript. We performed MD simulations using two different force fields in this study. The simulations using the Amber99SB*-ILDN [151, 152] force field and the resulting data analysis were part of my master's thesis and are included (Chapter 3.2) for completeness. During my PhD, I extended this by exploring the effects of phosphorylation using the CHARMM36 [153] force field. This extended study shows how phosphorylation impacts proteins structural dynamics and stability compared to the mutation at the same site.

The second case-study (Chapter 3.3) is based on the publication titled "Epistatic interactions promote persistence of NS3-Q80K in HCV infection by compensating for protein folding instability" [14], of which I was a joint first author. My contribution to this work involved performing MD simulations and analyzing the resulting data, which were critical in elucidating the role of epistatic interactions in the persistence of the NS3-

Q80K mutation in hepatitis C virus (HCV). The sub-chapter presents a collaborative effort with Dr. Christoph Welsch's experimental group at the University Hospital Frankfurt. Only the results I contributed to the publication have been taken from the publication and discussed here. Based on the research outcomes, I have also summarized the experimental results for completeness and included relevant figures from the publication in Chapter 3.3.2.2.

3.1 UNIFIED METHODOLOGY FOR MD SIMULATIONS OF KIT PROTEIN AND NS3 PROTEASE MODELS

In this section, we provide details of the methodology for MD simulations applied to the KIT protein (Chapter 3.2) and NS3 protease (Chapter 3.3) models, reflecting the consistent and robust approach used across these studies. KIT, a stem cell factor receptor involved in cell differentiation, whose dysregulation leads to various types of cancer [154]. The NS3 protein, encoded by the HCV genome, is a multifunctional enzyme crucial for viral replication and assembly, functioning as both a protease and an RNA helicase [155]. By consolidating the methods applied to these two studies, we highlight the shared techniques and analysis strategies, emphasizing the versatility and reliability of our MD simulations in exploring the dynamics and structural perturbations caused by mutations and post-translational modifications studied in different proteins.

3.1.1 TARGET SELECTION

For both case studies, crystallographic structures were retrieved from the Protein Data Bank (PDB) [135] and were the foundation for modeling and simulation.

The KIT protein's wild-type (WT) inactive and active states were retrieved from the PDB (PDB ID of inactive state 1T45 [156] and active state 1PKG [157]). The water molecules in the crystal structures were retained for the MD simulations. The missing atoms in the crystal structure of the active state of the protein were added using MODELLER 9.17 [158, 159] and the phosphorylated tyrosine residues at positions 568 and 570 were modeled back to their unphosphorylated state. In silico substitution of

Tyr (Y) to Asp (D) at position 823 was performed using MODELLER with the corresponding WT structures as the template for both active and inactive states. Tyr (Y) at position 823 was also phosphorylated in both monoanionic (TP1) and dianionic (TP2) states using CHARMM-GUI [160–162]. Generated models of the KIT wild-type (WT), its mutant (MU) Y823D, and two phosphorylated versions (TP1, TP2) are referred to as KIT-A^{WT}, KIT-A^{MU}, KIT-A^{TP1}, KIT-A^{TP2} and KIT-I^{WT}, KIT-I^{MU}, KIT-I^{TP1}, KIT-I^{TP2} for active and inactive states, respectively.

The models for the NS3-Q80K protease mutants were created using FoldX [163] based on the structure of the NS3 protease-helicase complex from genotype 1b (PDB ID: 1CU1) [164], which is 92% identical to genotype 1a in the protease domain. The structure of the protein monomer was extracted for further analysis.

3.1.2 PREPARATION OF THE SYSTEMS

MD simulations were performed using the GROMACS software package version 5.1.4 [165] employing the Amber99SB*-ILDN [151, 152] and CHARMM36 [153] force fields.

The molecular systems KIT-A^{WT}, KIT-A^{MU}, KIT-I^{WT}, and KIT-I^{MU} were parameterized using the Amber99SB*-ILDN force field. In a separate set, all generated models were parameterized using the CHARMM36 force field, and these variants are further referred to as KIT-A^{*WT}, KIT-A^{*MU}, KIT-A^{*TP1}, KIT-A^{*TP2} and KIT-I^{*WT}, KIT-I^{*MU}, KIT-I^{*TP1}, KIT-I^{*TP2}. The molecules were centered in a cubic box with a 1.5 nm buffer under periodic boundary conditions, and the systems were explicitly solvated with TIP3P water molecules. Counterions [166] ClJ and NaJ were added for Amber99SB*-ILDN force field simulations and Cl and Na for CHARMM36 force field simulations to neutralize the overall charge (0.15 mol/L concentration). Energy minimization for each molecular system was performed using the steepest descent algorithm. A maximum of 50000 steps was performed until a maximum force of 1000 kJ mol^{-1} nm^{-1} was achieved. Following the energy minimization, each of the molecular systems was subjected to two consecutive steps of the equilibration procedure. At first, each system was maintained at a temperature of 310 K during the NVT ensemble for 100 ps with a time step of 2 fs, followed

by a 100 ps simulation in the NPT ensemble with a time step of 2 fs, maintaining the pressure at 1 bar to equilibrate the system.

The NS3-Q80K protease models (WT protease and mutants: A91S, A91T, S174N) were optimized, and MD simulations were performed using the Amber99SB*-ILDN force field. The simulation parameters, including energy minimization, equilibration, and production phases, adhered to the procedures established for the KIT protein simulations using the same Amber99SB*-ILDN force field.

3.1.3 PRODUCTION OF TRAJECTORIES

In both studies, production runs were conducted to analyze the dynamics of the protein models, and coordinates were recorded at 100 ps intervals.

A total of 64 100 ns-long simulations were performed, with eight replicas for each system parameterized by the Amber99SB*-ILDN force field and four replicas for each system parameterized by the CHARMM36 force field. The temperatures of solute and solvent were separately coupled to the velocity rescale thermostat (modified Berendsen thermostat) [167] at 310 K with a relaxation time of 0.1 ps. The pressure was maintained at 1 atm by isotropic coordinate scaling with a relaxation time of 5 ps using Parrinello-Rahman barostat [168]. A time step of 2 fs was used to integrate the equations of motion based on the leap-frog algorithm [169]. Lennard-Jones interactions were set to a cut-off of 1.4 nm, and the Particle Mesh Ewald (PME) method [170] was used to treat long-range electrostatic interactions. All bonds were constrained using the P-LINCS algorithm [171].

For the NS3-Q80K protease models, 28 simulations of 100 ns each were performed, with four replicas for each of the NS3-Q80K models (WT protease and NS3-Q80K mutants: A91S, A91T, and S174N). The same production parameters were used as in the KIT protein simulations, including recording coordinates at 100 ps intervals and maintaining system conditions throughout.

3.1.4 ANALYSIS OF THE TRAJECTORIES

The resulting trajectories from the simulations were analyzed using various tools, including those from the GROMACS package. For both

studies, the first 10 ns were considered part of the equilibration process and were therefore excluded from the analysis. The last 90 ns of each trajectory were retained for further analysis, except for hydrogen bond (H-bond) calculations in KIT.

The secondary structure profiles were calculated using the *gmx do_dssp* program, which uses the DSSP algorithm [172], available in GROMACS. The calculation was performed over the concatenated trajectories of each system. The consensus secondary structure was defined as the type of secondary structure most prevalent at a given position over the whole simulation time (only α-helices and β-strands were considered) and visualized using the Biotite [173] package in Python. The secondary structure elements 3_{10}-helix, pi-helix, bend, turn, bridge, and coil were jointly marked as coil.

Two characteristic distances were monitored over the MD simulations of each system of KIT proteins to study the coupling between JMR and KD. The distance d1 between the centroid of the JM-B region (residues 547-559 as C1) and the centroid of the residues in the N-lobe (residues 582-692 as C1′) and the distance d2 between the centroid of the JM-S region (residues 560-570 as C2) and the centroid of the residues in the C-lobe (residues 763-935 as C2′). From the MD trajectories, the H-bonds between key residues were calculated using the program *gmx hbond* available in GROMACS. H-bonds were defined with a DHA angle cutoff of 120° and a donor-acceptor distance cutoff of 3.5 Å. The mean and standard error of the mean were calculated by a custom script written in R [174].

For the molecular systems parameterized by the CHARMM36 force field, we conducted RMSD, RMSF, secondary structure, H-bonds, and principal component analysis (PCA) for an in-depth analysis. These analyses provided a comprehensive understanding of KIT protein conformations' structural dynamics and stability under different conditions. Through this approach, we identified the key conformational changes and interactions that occur due to phosphorylation at Y823, highlighting its significant impact on the overall behavior of the protein.

For the NS3-Q80K protease models, including the WT protease and mutants (A91S, A91T, and S174N), we performed RMSD, RMSF, and H-bond analyses alongside residue interaction networks (RINs). The first ten ns were removed as part of the equilibration process, and the

rest of the 90 ns from each of the simulated protease models were concatenated with their replicas. From each concatenated trajectory, 181 frames were extracted at equal time steps, and RINs were created using the RINerator [175], a tool for analyzing molecular interactions between individual amino acids, offering insights into structure-function relationships. Custom scripts in Python were then developed and used to investigate the number of contacts for each position of interest.

3.1.5 PRINCIPAL COMPONENT ANALYSIS

Principal component analysis (PCA) is a statistical method for reducing the dimensionality of a complex system, such as the dynamic conformational space explored during MD simulations. By reducing the system's dimensionality, PCA allows for the extraction and interpretation of the most significant collective motions that contribute to the overall behavior of the protein.

Before performing PCA, the concatenated trajectories from replica simulations were superimposed to minimize the variance over the ensemble [176], ensuring that the analysis focuses on the protein's intrinsic motions and dynamics. The key idea of PCA is to identify the significant eigenvectors that define the dominant collective motions within the system. These eigenvectors, or principal components, represent the directions in which the system exhibits the most significant structural fluctuations.

The calculation was performed using the *gmx covar* module of GROMACS. The overlap between the first ten modes of each trajectory was calculated using the *gmx anaeig* module of GROMACS. This overlap provides insights into how different conditions, such as the Y823D mutation or phosphorylation at Y823, affect the protein's dynamic behavior. The perturbations of the systems can be described in terms of only a few principal components by ordering the eigenvalues of the diagonalized covariance matrix in a descending fashion. Thus, PCA helps extract the large-scale motions from MD trajectories by isolating the dominant modes of internal motion.

3.1.6 MUTUAL INFORMATION

Calculating mutual information between individual pairs of amino acids allows us to identify whether the changes in the pairs of conformational distributions of these amino acids are correlated linearly or non-linearly. It is assumed that such correlated changes can cause perturbations to the energy landscape. Both backbone and side chain torsion angles (ϕ, ψ, and χ) were used to study the correlated motion between pairs of residues as they are assumed to contain the functionally relevant perturbations and conformational changes [177]. The Mutual Information (MI) between them was estimated using the MutInf method [177], which allows us to capture significant concerted conformational changes of two residues as it focuses on torsion angles responsible for the low-frequency motions. We used a bin size of 24 degrees to obtain discrete distributions of the dihedral angles. The MI between pairs of residues is calculated as the individual sum of their entropies and subtracted from their joint entropy over adaptive partitioning. Each MI value was then compared to the background distribution of all MI values for all pairs of amino acids in the trajectories of the WT and mutant structures. A p-value threshold of 0.01 was applied to filter significant correlations.

To account for propagated error when comparing MI of WT and MU, bootstrap sets were created, following the procedure as described previously [178]. For this purpose, each dataset's torsion angles were extracted from individual trajectories using the *gmx g_chi* command in GROMACS and sampled with replacement n frames from a simulation of length n. This procedure was repeated ten times for each replica while preserving the same order across the different dihedral angles and simulation runs. The mean and standard deviation, μMI_{res} and $sdMI_{res}$, were calculated for every pair of correlated residues MI_{res} from the ten bootstrap sets. When comparing the mutual information in the wild-type and mutant complexes, $MI_{res_{WT}}$ and $MI_{res_{MUT}}$, only those residue pairs were retained, for which the following condition holds,

$$|\mu_{MI_{res_{MUT}}} - \mu_{MI_{res_{WT}}}| > \frac{\sqrt{sd^2_{I_{res_{MUT}}} + sd^2_{I_{res_{WT}}}}}{N} \tag{3.1.1}$$

Where N is the total number of independent replica simulations. To further eliminate the noise and weak interactions between correlated residues, a cutoff (MI > 0.8 kT for the inactive state and MI > 1.7 kT for the active state) was chosen, so pairs with a slight absolute difference between mean values were disregarded. When comparing the WT and MU simulations, this step was necessary to select only the correlated residues with largely different mean MI values.

3.1.7 PARTIAL LEAST-SQUARES REGRESSION

Partial least-squares regression (PLS) was applied to identify the collective modes of internal dynamics associated with an external order parameter of functional interest using the functional mode analysis tool [179]. For this analysis, the coordinates of the backbone atoms, backbone atoms without JMR region, and all protein atoms, excluding hydrogen atoms, were used to build the statistical models. The constants 0 and 1, corresponding to the trajectories in WT and MU simulations, were used as the read-out variable to be estimated.

To validate the generated statistical models, we applied the following adapted k-fold cross-validation (CV) technique for WT and MU trajectories separately in the active and inactive states, following the procedure described before [178]. The trajectories of WT and MU of the inactive state KIT protein were concatenated and superimposed to minimize the variance over the ensemble [176]. The resulting trajectory was divided into four equal parts. Three parts of the data with the labeled input containing equal parts from both WT and MU were then used in each iteration to train a model. Based on this, we made predictions for the last part. The final number of PLS modes/components was chosen based on the Pearson correlation calculated between the actual and the predicted values, with a compromise between the number of modes, the complexity, and the prediction quality using the "elbow method" [180, 181].

3.2 STRUCTURAL DYNAMICS OF KIT: MD SIMULATIONS OF Y823 PHOSPHORYLATION AND COMPARATIVE EFFECTS OF Y823D

Here, we discuss the role of tyrosine kinases (TKs) [182, 183] in signaling cascades and their potential as targets for anti-cancer drugs. Our focus is on the stem cell factor receptor KIT, a TK involved in cell differentiation, whose dysregulation leads to various types of cancer [154]. We investigated the effect of phosphorylation of Y823 using MD simulations with the CHARMM36 force field during my PhD. We compared it to the impact of the mutation Y823D, which was studied during my master's using the Amber99SB*-ILDN force field. This comparison shows how both phosphorylation and mutation at the same site affect the structural dynamics and stability of KIT, contributing to a comprehensive understanding of the protein's behavior in the context of drug resistance. Previous studies, such as the one by Agarwal *et al.* [154], demonstrated that Y823 phosphorylation is crucial for maintaining receptor activity, preventing internalization and degradation, and sustaining downstream signaling. My work extends this by comparing the dynamics of the wild-type, mutant, and phosphorylated KIT conformations. We found that phosphorylation at Y823 plays a critical role in stabilizing the active conformation of KIT, similar to the effects observed with the Y823D mutation. This study highlights how these modifications could influence KIT's response to inhibitors.

3.2.1 BACKGROUND: TYROSINE KINASES

The human genome contains 518 genes encoding kinases, of which 90 encode protein tyrosine kinases (TKs) [182, 183]. TKs are enzymes that phosphorylate a tyrosine residue of a target protein or phosphorylate their tyrosines, leading to conformational changes and, typically, activation of downstream signaling cascades. Thus, TKs function as "on" or "off" switches in many cellular processes.

3.2.1.1 CLASSIFICATION

The 90 TKs can be grouped into two classes: 58 belong to receptor tyrosine kinases (RTKs), and 32 are non-receptor tyrosine kinases (NRTKs) [184, 185]. The significant difference between these two classes is that the NRTKs, in contrast to RTKs, do not have the extracellular domain responsible for binding extracellular ligand molecules [186]. From here on, we will focus on RTKs.

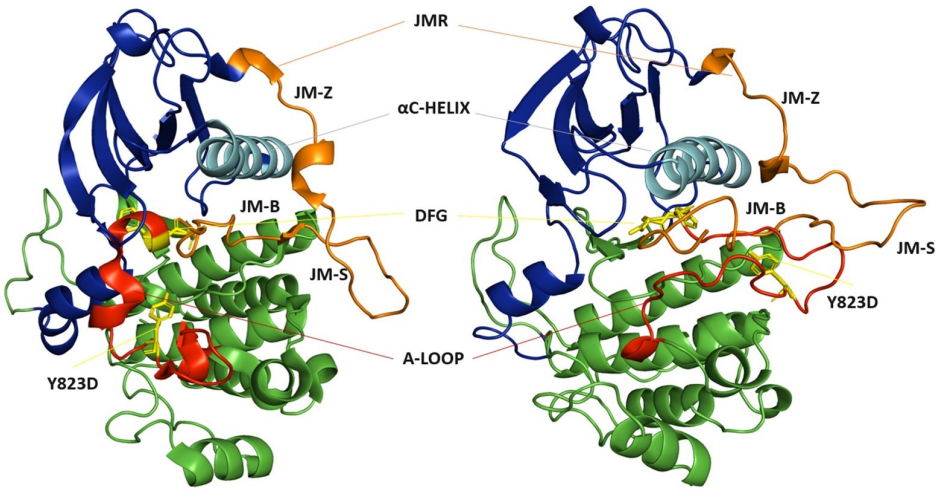

Figure 3.1: *Structure of KIT cytoplasmic region. 3D structures of the native receptors in the inactive state (PDB id:* 1T45 *[156]) and in the active state (PDB id:* 1PKG*[157]) are represented as cartoons. The N-lobe in blue, the C-lobe in green, the αC-helix in cyan, the A-loop in red, and the JMR in orange colors. The DFG motif and the mutation site Y823D are represented in the sticks (yellow).*

3.2.1.2 RECEPTOR TYROSINE KINASES (RTKS)

Cell activity changes in response to the signals from their surrounding environment. Most of the activities are controlled by extracellular signaling molecules. Transmembrane receptors often transduce these signals across the cell membrane. One such type of membrane protein is the RTK. RTKs are cell surface receptors for several growth factors, hormones, and cytokines [187, 188]. Their structural characterization divides RTKs into 20 subfamilies with a homologous domain specifying the catalytic TK function [189]. The RTK family consists of growth factor receptors, which include the platelet-derived growth factor receptor (PGDF-R), the epidermal growth factor receptor (EGFR), fibroblast growth factor (FGF), the vascular endothelial growth factor (VEGF), the

insulin receptor family among others [188–190]. The RTKs consist of a single transmembrane domain separating the intracellular tyrosine kinase domain from the extracellular domain, and the tyrosine kinase domain contains several conserved regions, including the ATP binding site that catalyzes the auto-phosphorylation and transphosphorylation of tyrosine residues [187, 188, 191, 192].

3.2.1.3 MECHANISM OF ACTIVATION AND SIGNALING

The RTK activation [187, 191, 192] involves binding a ligand to the RTK monomer, leading to the dimerization of the receptor. Upon binding a ligand to the extracellular domain, the receptor undergoes extensive conformational changes that induce and stabilize the receptor dimerization, stimulating the kinase activity in the intracellular domain and thus resulting in auto-phosphorylation of tyrosine residues. The intracellular TK domain has a bi-lobar structure, with an ATP binding cleft located between the N- and C-terminal lobes that catalyze the auto-phosphorylation and trans-phosphorylation of tyrosine residues and a kinase insert domain [192, 193]. The N-lobe (residues 582-692) is composed of antiparallel β sheets adjacent to the α-helix (αC-helix), and the C-lobe (residues 763-935) shows predominantly the α-helical structure (Figure 3.1) [194]. The C-lobe contains the G-helix that binds the kinase substrate and an activation loop (A-loop) (residues 810-835) that begins with a highly conserved DFG (residues 810-812) motif [194]. The major autophosphorylation sites are in the juxtamembrane region (JMR, residues 547-581) and the kinase domain (KD, residues 582-935) [193, 194]. The activated receptor's phosphorylated tyrosine residues now act as a binding site for proteins containing the Src homology 2 (SH2) and phosphotyrosine binding (PTB) domains [195–197].

In its unphosphorylated state, the intracellular TK domain exists in its inactive auto-inhibited conformation (Figure 3.1) [194, 198, 199]. As part of the activation process [194, 200–202] upon ligand binding to the extracellular domain, the A-loop adjacent to the active site in the inactive state switches from its auto-inhibitory position to a more open form. During this process, the DFG motif at the N-terminus of the A-loop flips its side chain away from the ATP binding site, thus allowing for the binding of the ATP and Mg^{2+} cofactors. Following these structural changes, the JMR unwinds from its buried position in the TK domain

to a solvent-exposed position, and the αC-helix undergoes orientational changes, breaking the contacts with the JM-B fragment of the JMR. Along with JM-B, the JM-Z fragment also blocks the αC-helix, which regulates the catalytic activity of the kinases and prevents the A-loop from adopting an active conformation, restricting the inter-lobe flexibility [203–205].

3.2.1.4 TYROSINE KINASES AS TARGETS FOR ANTICANCER AGENTS

In 1984, the first connection between a viral oncogene, a mutated RTK, and human cancer was established [206]. Since then, it has been well established that abnormal signaling by RTKs is critically involved in human cancer [188]. The profuse knowledge of the structure and activation mechanism of RTKs and the variations of TK signal transduction pathways in proliferative disorders led to the idea that tyrosine kinase inhibitors (TKIs) could have anticancer effects [183, 188, 207, 208]. As a result, the development of target-specific TKIs and new anticancer drug discovery has become a hot area of anticancer research.

Most TKIs are small hydrophobic compounds that can rapidly reach their specific intracellular targets and inhibit the activation of the related TKs [183, 209, 210]. Unfortunately, the patients who gained remarkable benefits from the TKI therapy showed increasing evidence of acquired resistance [211–213]. It is primarily due to the acquired secondary drug-resistance mutations that change the protein conformation and alter the drug binding site, leading to therapy failure and cancer relapse [214–216]. These mutations may alter the tightly controlled functions of the protein, including ligand binding, conformational transitions, and allosteric regulation, inducing resistance to several first and second-line drugs [183, 193, 202]. However, the exact molecular mechanisms of the changes caused by such mutations are still not well understood.

3.2.1.5 KIT PROTEIN TYROSINE KINASE

KIT is a stem cell factor (SCF) receptor, also known as proto-oncogene receptor tyrosine kinase, a member of class III of RTKs. The activation of KIT proceeds similarly to other RTKs: an SCF molecule binds to a KIT monomer in the extracellular domain, leading to dimerization,

activation, and autophosphorylation of tyrosine residues in the KD [157, 217–219]. During this process, the key regulatory regions in the TK domain undergo extensive conformational changes. A gain-of-function or an acquired secondary drug resistance mutation in the TK domain can activate the protein without the ligand [182, 183, 192, 203]. D816V is one such mutation studied extensively using MD simulations [194, 219–221]. It has been shown that this mutation in the inactive state disrupts the communication between the A-loop and the JMR [194, 220] and causes a partial folding of conserved 3_{10} helices in the A-loop, leading to changes in the local hydrogen-bond (H-bond) network [219], the global structural reorganization of the JMR, and constitutive activation of the protein [11]. KIT harboring this mutation is resistant to several drugs, including imatinib and sunitinib [12, 222, 223]. Y823D is a similar gain-of-function or a secondary mutation reported in several clinical cases [224–227]. Y823D can activate the PI3K/AKT, RAS-ERK, and JAK/STAT pathways, leading to tumorigenesis by inducing cell proliferation, growth progression, or migration [12, 154, 226]. This mutation is associated with various forms of cancer, including gastrointestinal stromal tumors, testicular seminomas, melanogenesis, and hematopoiesis [11–13]. Yet, very little is known about the impact of this mutation on the protein conformational changes crucial for activation and ligand binding.

It has been reported that Y823D can cause stabilization of the active conformation of the protein and that it is also responsible for the loss of sensitivity to drugs in metastatic tumors [228]. This mutation occurs at the only tyrosine residue in the A-loop of the KD, which is the last residue that undergoes autophosphorylation [154, 157, 229]. Although this tyrosine residue is not essential for the initial activation of KIT, it plays a crucial role in stabilizing the active conformation of the receptor. This stabilization is important for downstream signaling and receptor stability and influences drug sensitivity and cellular outcomes [154, 223, 227, 230]. It is also critical for regulating kinase activity and cell survival and proliferation [154, 157, 204, 223]. In this study, we extend our understanding of KIT protein function by emphasizing the role of tyrosine phosphorylation in Y823. We explored the impact of the Y823D mutation and tyrosine phosphorylation on protein conformation using MD simulations. Our goal was to elucidate the differences in

dynamics between various protein states, allowing us to identify the conformational changes essential for activation and ligand binding.

3.2.2 RESULTS

In this study, we analyzed the KIT molecular systems' structural features and internal dynamics to investigate the differences between the conformations induced by the gain-of-function mutation Y823D and the two variants of Y823's phosphorylation. We examined these differences to understand their impact on KIT's dynamics and stability, focusing on the role of Y823 phosphorylation in modulating the receptor's behavior.

I want to note that simulations using the Amber99SB*-ILDN force field were conducted as part of my master's thesis, which primarily focused on the mutation Y823D. These results are included in this study for completeness and comparative analysis. However, this work is extended by incorporating the results of Y823 phosphorylation using the CHARMM36 force field. This extension explains how Y823 phosphorylation influences KIT dynamics, stability, and function.

3.2.2.1 MOLECULAR FLEXIBILITY OVERVIEW

The root mean square deviations (RMSDs) were calculated for the backbone atoms and the key structural elements (αC-helix, JMR, and A-loop) (Figures 3.2A and 3.3A) of the kinase domain (KD) to ascertain the time evolution of the structure. The backbone RMSD profiles of the inactive and active states show that they conform closely to their initial crystallographic structures. The RMSD mean values ranged from 0.16 to 0.25 nm for the inactive state and 0.36 to 0.52 nm for the active state in trajectories resulting from simulations using both Amber99SB*-ILDN and CHARMM36 force fields. RMSD profiles show that the conformational drift of the JMR in the inactive state of the protein is larger than the other KIT regions. In the active state, along with JMR, the A-loop also had a large RMSD, whereas the αC-helix is rigid in both states.

The protein flexibility was estimated by the Root Mean Square Fluctuations (RMSFs) averaged over time for every residue. The RMSF values of the backbone atoms (Figures 3.2B and 3.3B) were comparable between different systems, ranging from 0.1 to 0.3 nm in the inactive state and 0.1

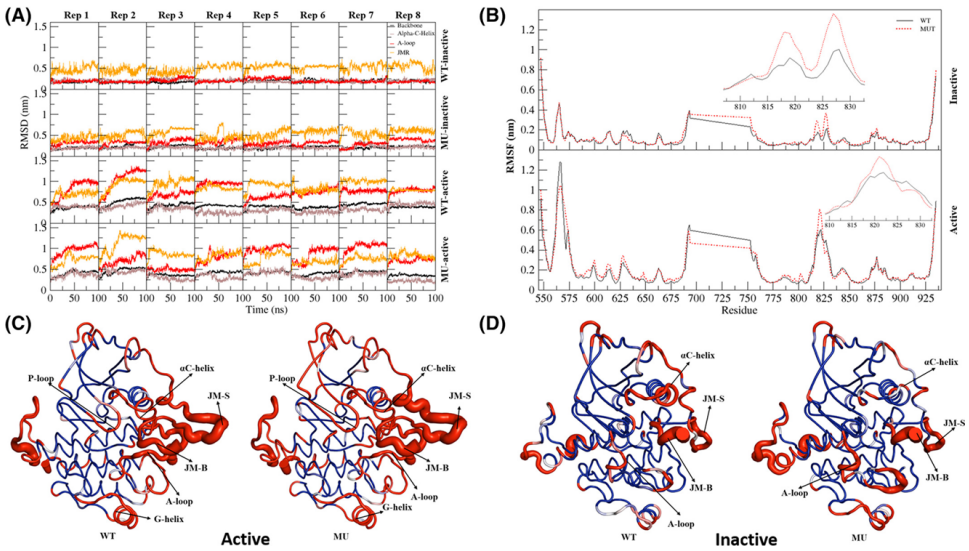

Figure 3.2: *(A) The RMSD values were calculated for backbone atoms (black), αC-helix (brown), A-loop (red), and JMR (orange) from all eight trajectories of MD simulations of KIT^{WT} and KIT^{MU} in both the states (active and inactive). (B) The RMSF values computed on the backbone atoms of the concatenated trajectory of MD simulations of KIT^{WT} (black) and KIT^{MU} (red). The RMSF values of the A-loop are given in the insert. The average conformation of the (C) active state KIT and (D) inactive state KIT is represented as tubes. The highly flexible regions are shown in red, highly stable residues in blue, and the intermediate flexible residues in white. The tube size is proportional to the observed RMSF values in those regions.*

to 0.4 nm for the active state in trajectories resulting from simulations using both Amber99SB*-ILDN and CHARMM36 force fields. From the RMSF plot Figure 3.3B), we can also observe how the MU and phosphotyrosine simulations converge, showing that the Y823D mutation mimics the presence of phosphotyrosine. Both variants of the phosphotyrosine behave very similarly in the simulation.

Further, we can also observe that the WT and MU simulations of both active and inactive states performed using two different force fields agree very well (Figures 3.3B and 3.4).

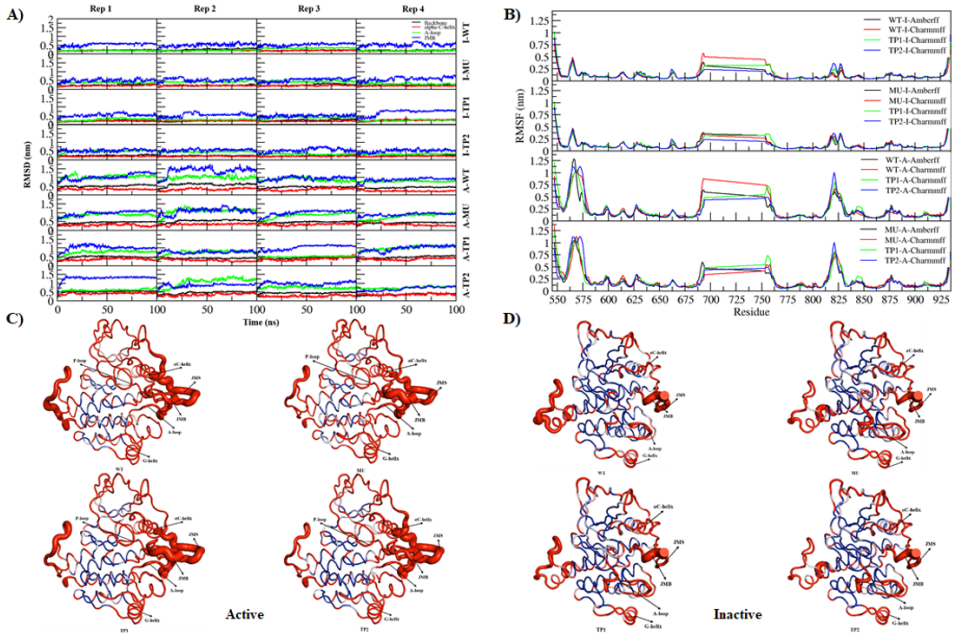

Figure 3.3: *(A) The RMSD values calculated for backbone atoms (black), αC-helix (red), A-loop (green), and JMR (blue) from all four trajectories of MD simulations of KIT-A*^{WT}, KIT-A*^{MU}, KIT-A*^{TP1}, KIT-A*^{TP2} and KIT-I*^{WT}, KIT-I*^{MU}, KIT-I*^{TP1}, KIT-I*^{TP2}. (B) The RMSF values were computed on the backbone atoms of the concatenated trajectories resulting from simulations using Amber99SB*-ILDN and CHARMM36 force fields. The average conformation of the (C) active state KIT and (D) inactive state KIT is represented as tubes. The highly flexible regions are shown in red, highly stable residues in blue, and the intermediate flexible residues in white. The tube size is proportional to the observed RMSF values in those regions.*

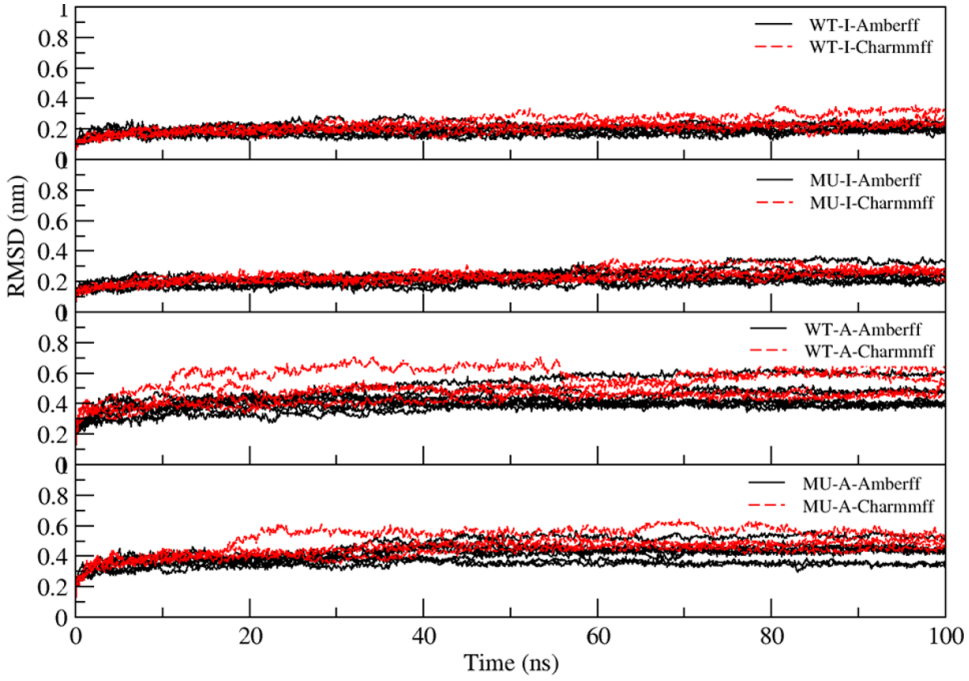

Figure 3.4: *The RMSD values were calculated for backbone atoms of inactive and active state WT and MU individual replicas resulting from simulations using Amber99SB*-ILDN (black) and CHARMM36 (red) force fields.*

3.2.2.2 SECONDARY STRUCTURE ANALYSIS

Upon analyzing the trajectories resulting from simulations using both Amber99SB*-ILDN and CHARMM36 force fields, we observe that the mutation does not significantly alter the secondary structure of JMR in both states of the protein (Figures 3.5 and 3.6) compared to its WT. Still, we observe slight changes in the secondary structure type during our simulations.

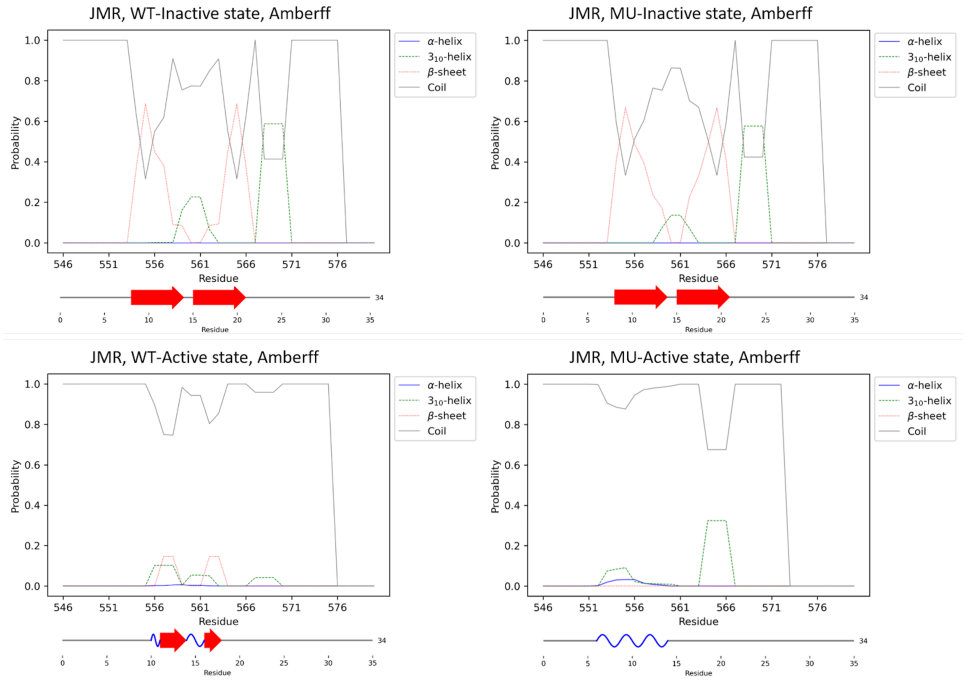

Figure 3.5: *Secondary structure assignments of KIT JMR averaged over 720 ns MD simulations of KIT-I^{WT}, KIT-I^{MU}, KIT-A^{WT}, and KIT-A^{MU}. Only the major secondary structure elements (SSE) are shown. In the consensus representation, the coil SSE includes 3_{10}-helix, pi-helix, bend, turn, bridge, and coil; helices are shown in blue, sheets in red, and coils in gray.*

The mutation Y823D is located in the A-loop and induces local fluctuations in the inactive state of the protein. Unlike other secondary mutations such as D816V/H/N/Y [11, 220] in the A-loop, the mutation Y823D does not destabilize the 3_{10}-helix in the segment 817-819 in the trajectories resulting from Amber99SB*-ILDN force field (Figure 3.7). In contrast, almost all secondary structure elements in the A-loop are lost in the active state. The 3_{10}-helix is better retained in the MU simulation with the CHARMM36 force field, whereas in KIT-I^{*TP1} and KIT-I^{*TP2} trajectories resulting from the CHARMM36 force field (Figure 3.8), we observe very little secondary structure in the A-loop. The 3_{10}-helix in

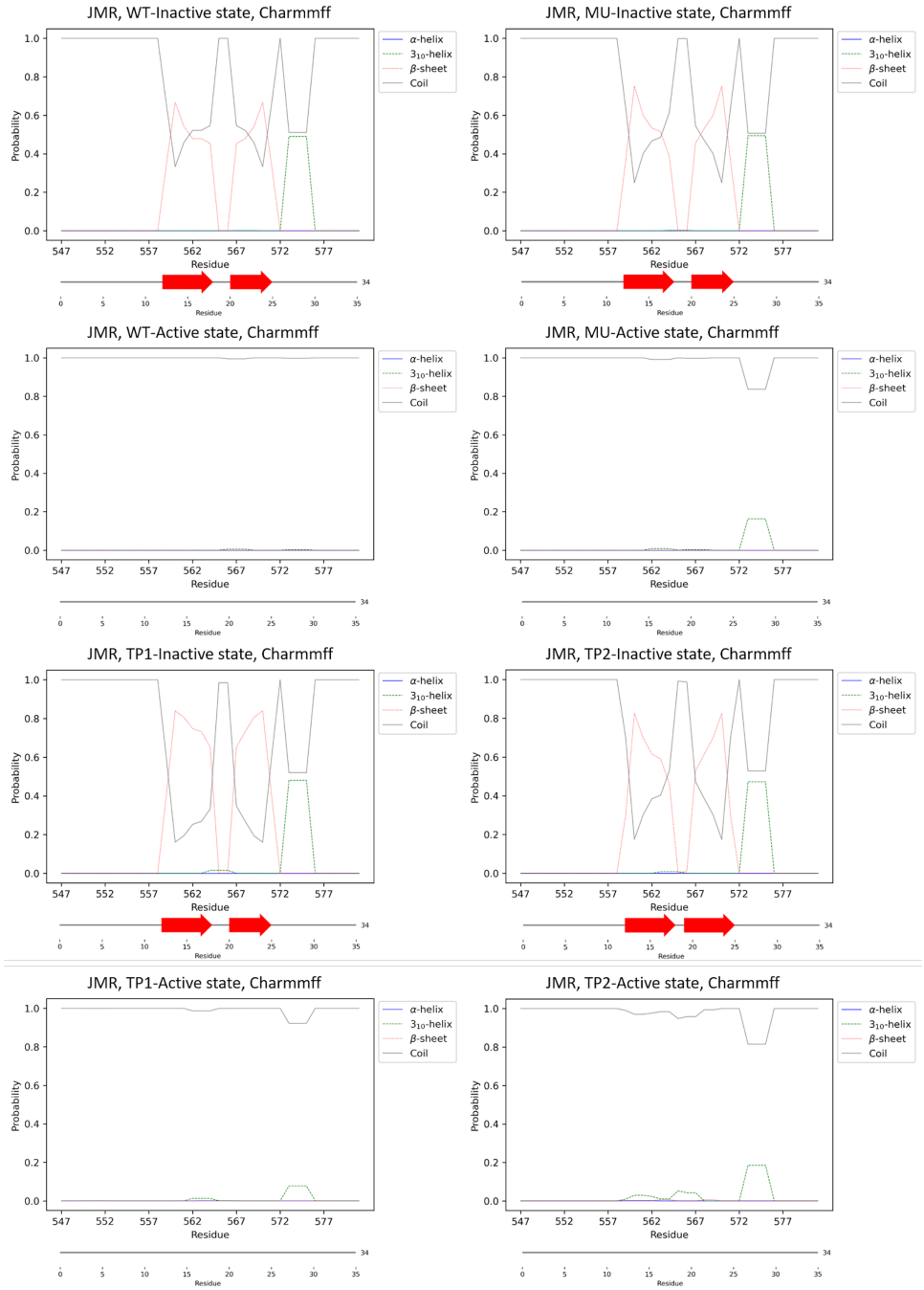

Figure 3.6: *Secondary structure assignments of KIT JMR averaged over 360 ns MD simulations of KIT-I*^{WT}, KIT-I*^{MU}, KIT-A*^{WT}, KIT-A*^{MU}, KIT-I*^{TP1}, KIT-I*^{TP2}, KIT-A*^{TP1}, KIT-A*^{TP2}. Only the major secondary structure elements (SSE) are shown. In the consensus representation, the coil SSE includes 3_{10}-helix, pi-helix, bend, turn, bridge, and coil; helices are shown in blue, sheets in red, and coils in gray.*

the A-loop is only observed in the inactive state of the protein, and destabilization of this segment has been reported to be associated with the disruption of its integrity [11]. Such destabilization of key regions in the A-loop is associated with significantly contributing to the loss of sensitivity to drugs [11].

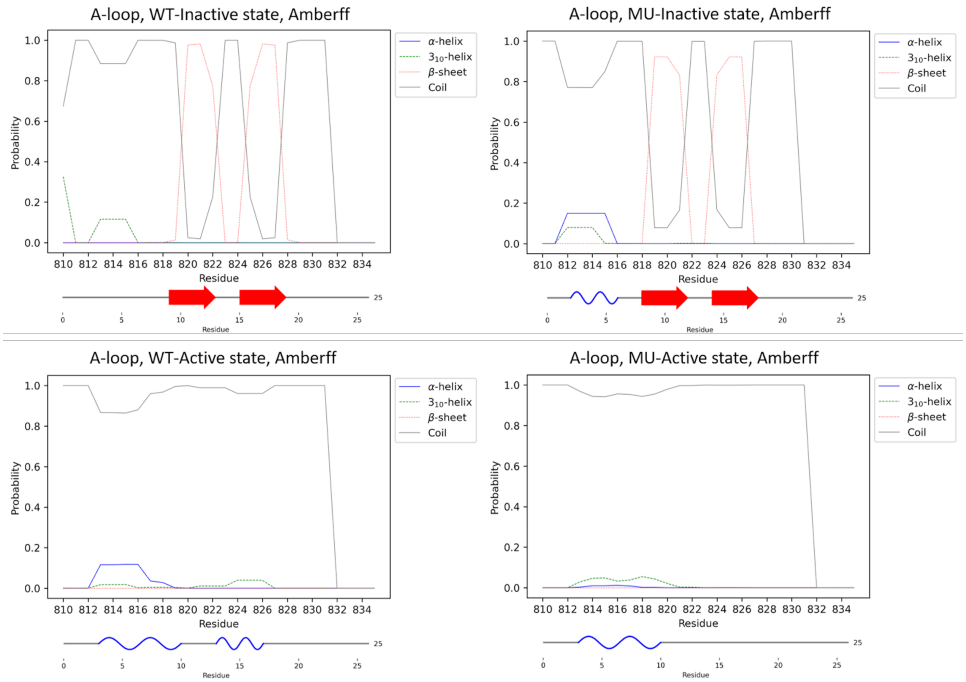

Figure 3.7: *Secondary structure assignments of KIT A-loop averaged over 720 ns MD simulations of KIT-I^{WT}, KIT-I^{MU}, KIT-A^{WT}, and KIT-A^{MU}. Only the major secondary structure elements (SSE) are shown. In the consensus representation, the coil SSE includes 3_{10}-helix, pi-helix, bend, turn, bridge, and coil; helices are shown in blue, sheets in red, and coils in gray.*

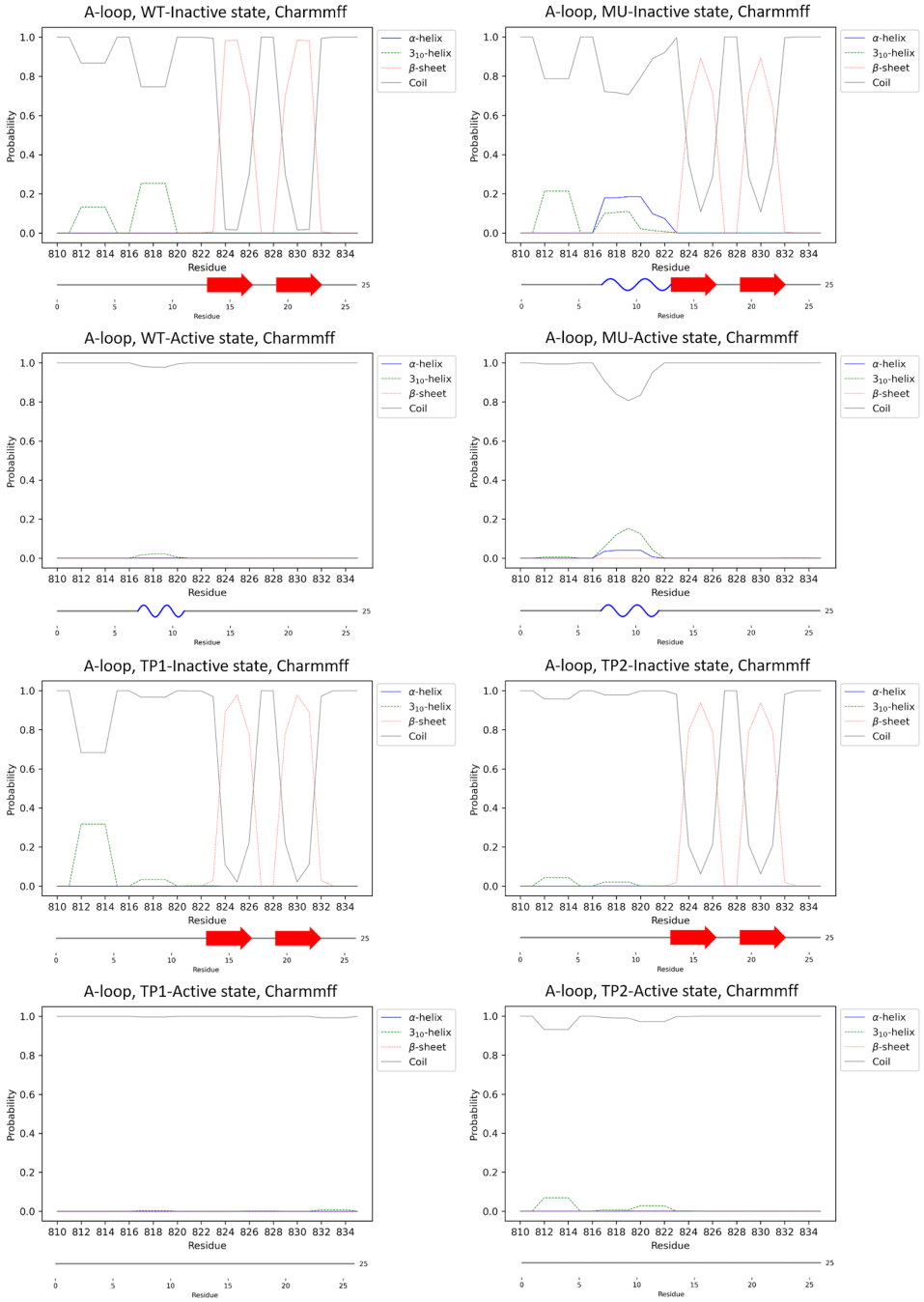

Figure 3.8: *Secondary structure assignments of KIT A-loop averaged over 360 ns MD simulations of KIT-I*^{WT}, KIT-I*^{MU}, KIT-A*^{WT}, KIT-A*^{MU}, KIT-I*^{TP1}, KIT-I*^{TP2}, KIT-A*^{TP1}, KIT-A*^{TP2}. Only the major secondary structure elements (SSE) are shown. In the consensus representation, the coil SSE includes 3_{10}-helix, pi-helix, bend, turn, bridge, and coil; helices are shown in blue, sheets in red, and coils in gray.*

3.2.2.3 DYNAMIC BEHAVIOR OF RECEPTORS

The effect of the mutation on the collective dynamic behavior of various KIT structural elements is markedly different in the active and inactive states. The computed scalar products between the first 10 PCA modes of KIT inactive state proteins indicate that their collective motions differ (Figure 3.9A). It is interesting to compare the WT and MU protein dynamics in each state.

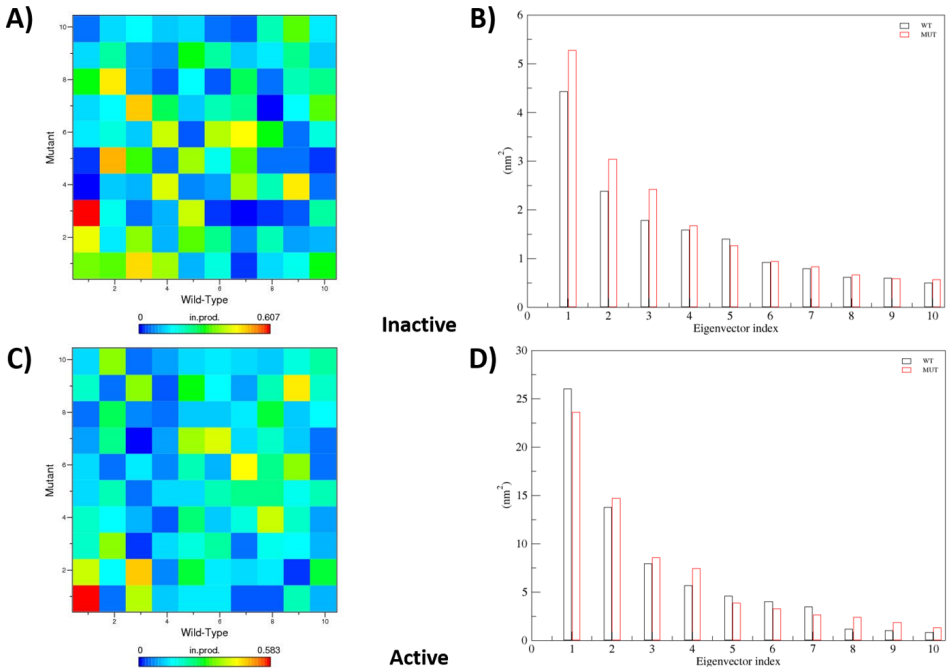

Figure 3.9: *Eigenvector and eigenvalue comparison of backbone PCA of KIT-I^{WT} and KIT-I^{MU}(A-B), KIT-A^{WT} and KIT-A^{MU} (C-D).*

When the trajectories of KIT-I^{WT} and KIT-I^{MU} are projected into the subspace spanned by their first two PCs (Figure 3.10A), one can observe a significant overlap of the WT and MU parts of the trajectories and a considerable shift along the PC1 axis. Modes 1 and 2 of KIT-I^{MU} display a slightly higher amplitude than KIT-I^{WT} (Figure 3.9), indicating increased protein flexibility upon mutation. These modes correspond to atomic motions (Figure 3.10B, C) of JMR coupled to KD deformations in the N-lobe, αC-helix, residues 626-631 preceding the αC-helix, orientational changes in the A-loop and the G-helix in the C-lobe. Principal modes of KIT-I^{MU} describe atomic motions of JM-B coupled with a twist motion of the N-lobe and a displacement of the A-loop and G-helix in the C-lobe. This displacement/outward motion of the A-loop is not characteristic of

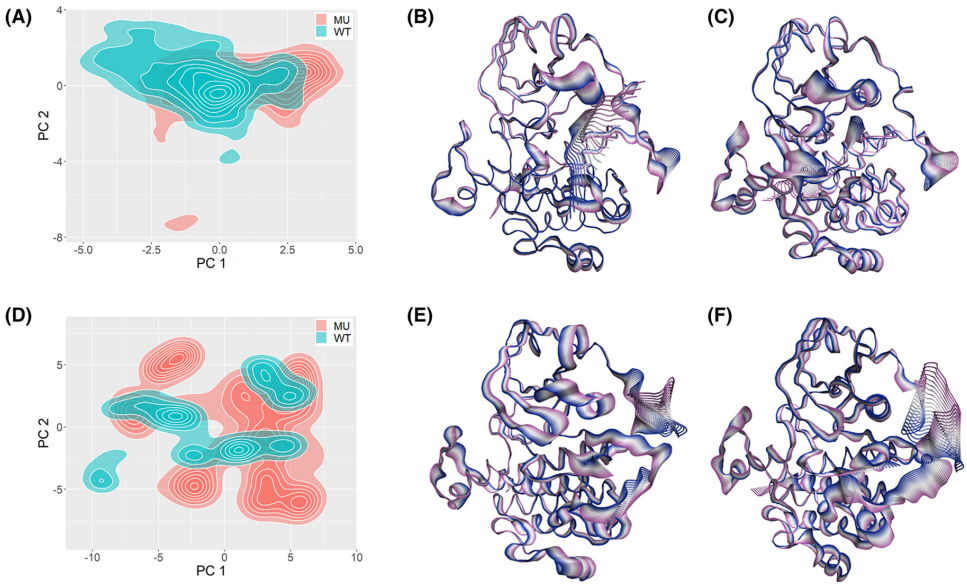

Figure 3.10: *PCA of KIT inactive and active states. The atomic motions are shown as an interpolation between the extremes along the first principal component of the (B, C) inactive and (E, F) active states. The conformation distribution of different states (WT: cyan, MU: red) of KIT is projected onto the subspace defined by the first two principal components (PC1 and PC2) of the concatenated WT and MU trajectories in the active (A) and inactive states (D).*

the active conformation. The RMSF analysis along the individual modes (eigenvectors) reveals that the JMR and A-loop portions dominate the total backbone fluctuations, kinase insert domain, and minor perturbations of the αC-helix (data not shown).

Mapping KIT-A^{WT} and KIT-A^{MU} into the subspace spanned by the first two principal components of their concatenated trajectories shows these trajectories occupy very different regions (Figure 3.10D). Analysis of the first eigenvectors of KIT in the active state indicates a good agreement between the first principal mode of KIT-A^{WT} and KIT-A^{MU}, but not so for the second mode (Figure 3.9C). Accordingly, the WT trajectory occupies a region similar to the MU trajectory along PC1 but differs much along PC2. The first eigenvalues for both WT and MU trajectories in the active state are much higher than those for the inactive state, indicating generally higher protein flexibility in the active state (Figure 3.9D). In KIT-A^{WT}, the first mode is associated with displacement of JMR, A-loop, and unwinding twist motion of the N-lobe (Figure 3.10E-F). At the same time, the KIT-A^{MU} experiences a conformational change in the JMR; its A-loop curtails from its elongated conformation, and an opposite twisted motion along an axis passing between the middle of the

N- and C-lobe happens. The RMSF analysis along the first eigenvector shows that the JMR, kinase insert domain, and the A-loop contribute the most to the backbone fluctuations in KIT-A^{WT} and KIT-A^{MU} (data not shown).

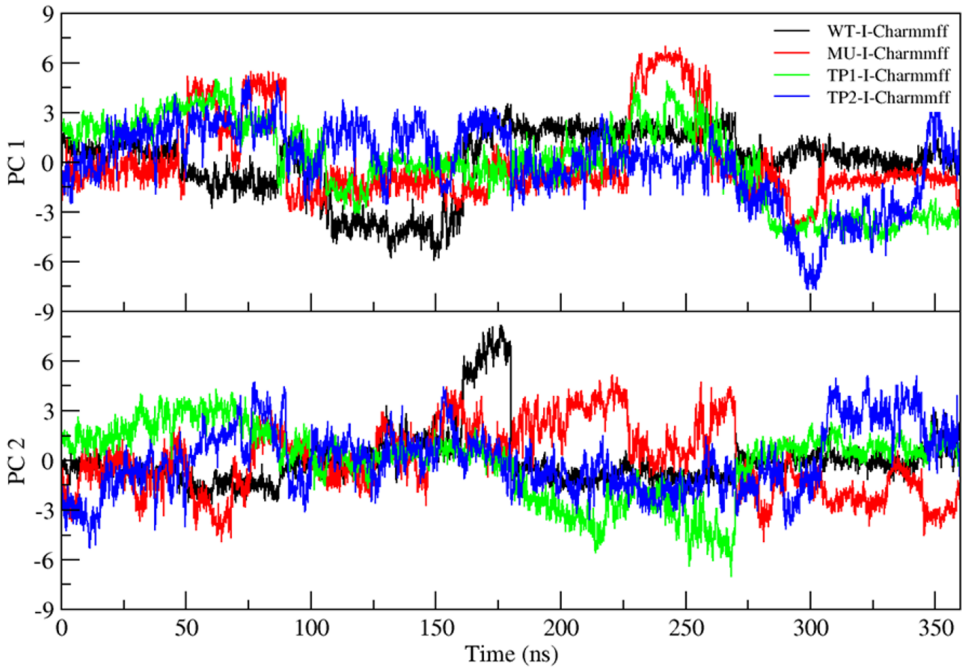

Figure 3.11: *Time series projection of snapshots of KIT-I^{*WT} (black), KIT-I^{*MU} (red), KIT-I^{*TP1} (green), and KIT-I^{*TP2} (blue) simulations onto the first two principal components (PC1 and PC2).*

PCA on the inactive state trajectories from CHARMM36 force field simulations confirms the similarity in the conformational distribution of the mutant and the phosphotyrosine simulations based on the projection onto the subspace defined by the first two PCs (PC1 and PC2) (Figure 3.11).

3.2.2.4 COUPLING BETWEEN JMR AND KD IN RECEPTORS

To check the possible coupling between JMR and KD in both protein states, their relative positions were characterized using two geometrical distances, d1 and d2. Monitoring these distances over the MD trajectories indicates that the d1 and d2 distributions (Table 3.1) in the inactive and active states of the KIT are very similar for both WT and MU. This demonstrates that the JMR is tightly coupled to the KD in both KIT conformations. The same can be observed in the PCA analysis; we observed that the A-loop is responsible for the dominant fluctuations

during the simulations of the active state (Figures 3.10E-F). Apart from these fluctuations, no major conformational changes are observed, and the computed d2 distance deviation between KIT-A^{WT} and KIT-A^{MU} is also negligible.

	KIT-I^{WT}	**KIT-I^{MU}**	**KIT-A^{WT}**	**KIT-A^{MU}**
d1	1.92 ± 0.01	1.92 ± 0.01	2.05 ± 0.04	2.08 ± 0.05
d2	2.3 ± 0.02	2.3 ± 0.02	3.05 ± 0.11	3.03 ± 0.09

Table 3.1: *Average distance between centroids of (d1) JM-B and N-lobe and (d2) JM-S and C-lobe. ± indicates standard error across independent simulations.*

The frequency of H-bonds between key residues that maintain the conformation of the KIT protein and the residues that are known to be involved in establishing the allosteric communication between JMR and KD, such as H-bond between D792 and Y823 [11, 219, 220, 231] was measured (Tables 3.2, 3.3, and 3.4). In KIT-I^{WT}, JMR binds to the C-lobe of KD through stable H-bonds V560...N787, K558...I789, Y568...F848 and Y570...Y846. These H-bonds are preserved in KIT's mutated and phosphorylated structures, indicating a solid coupling and stabilization of the JMR attachment to KD and the internal JMR contacts. These data are consistent with the data from the coupling analysis between JMR and KD.

The signal from the A-loop to the JMR is propagated through the C-loop, with Y823 as a critical intermediate residue [194]. The corresponding H-bonds are affected by the Y823D mutation and phosphorylation. The H-bond between residues Y823 and D792 is wholly lost in KIT-I^{MU}, KIT-I^{*MU}, KIT-I^{*TP1}, and KIT-I^{*TP2} due to the changed properties of the residue at position 823. The interaction between D792 and N797 is also observed less frequently. However, the H-bond between D792 and H790 is not affected. A decreased communication to the distant KIT regulatory elements [194] accompanies such modification in the local interaction network of the C-loop. These observations suggest that the allosteric communication in the KIT inactive state between the A-loop and JMR is disrupted by the mutation Y823D in the A-loop. A beta-turn motif (residues 820-823) supported by an H-bond D820...N822 also disappears in the mutated and phosphorylated structures. It is known that the H-bond D820...N822 stabilizes this beta-turn motif in the protein inactive state, and disruption of this H-bond results in the unfolding of the

	KIT-I^{WT}	KIT-I^{MU}	KIT-A^{WT}	KIT-A^{MU}
JMR...C-lobe				
V560...N787	1.067 ± 0.01	1.08 ± 0.01	0.001 ± 0	0.008 ± 0.01
K558...I789	1.80 ± 0.03	1.87 ± 0.03	0	0.25 ± 0.25
Y568...F848	0.29 ± 0.6	0.35 ± 0.12	0.04 ± 0.04	0
Y570...Y846	1.23 ± 0.23	1.09 ± 0.24	0	0
E640...F811	0	0	0.71 ± 0.14	0.55 ± 0.13
JMS				
V569...E561	1.02 ± 0.06	1.21 ± 0.16	0	0
A-loop				
V824...L831	2.022 ± 0.01	1.9 ± 0.06	0	0
R815...Y823	0.89 ± 0.22	0.09 ± 0.04	0.32 ± 0.16	0.733 ± 0.3
D820...N822	2.68 ± 0.08	1.65 ± 0.21	0.97 ± 0.22	0.98 ± 0.23
A-loop...C-lobe				
Y823...R796	1.254 ± 0.23	1.1 ± 0.46	0	0
Y823...Y846	0	0	0.22 ± 0.08	0.42 ± 0.2
C-loop				
D792...H790	1.931 ± 0.01	1.93 ± 0.02	1.53 ± 0.12	1.7 ± 0.07
H790...N797	0	0.13 ± 0.1	0.15 ± 0.1	0.16 ± 0.1
D792...N797	2.088 ± 0.1	1.83 ± 0.18	1.4 ± 0.16	1.53 ± 0.22
C-loop...A-loop				
D792...Y823	1.20 ± 0.19	0	0	0
C-loop...C-lobe				
N797...C809	0.56 ± 0.12	0.87 ± 0.15	0.85 ± 0.14	0.55 ± 0.17
C-lobe				
L799...K807	2 ± 0	2 ± 0	2 ± 0	2 ± 0

Table 3.2: *Average H-bond number between residues in Amber99SB*-ILDN force field simulations. ± indicates standard error across independent simulations.*

	KIT-I*^{WT}	KIT-I*^{MU}	KIT-A*^{WT}	KIT-A*^{MU}
JMR...C-lobe				
V560...N787	1.03 ± 0	1.11 ± 0.02	0	0
K558...I789	1.54 ± 0.07	2 ± 0.16	0	0.01 ± 0.01
Y568...F848	0.23 ± 0.09	0.59 ± 0.14	0	0
Y570...Y846	0.3 ± 0.12	0.16 ± 0.05	0	0
E640...F811	0	0	0.85 ± 0.14	0.73 ± 0.2
JMS				
V569...E561	1.13 ± 0.28	1.11 ± 0.09	0	0
A-loop				
V824...L831	2.02 ± 0.01	1.33 ± 0.08	0	0
R815...Y823	0.79 ± 0.39	0	0.05 ± 0.03	0.16 ± 0.04
D820...N822	2.24 ± 0.05	0.99 ± 0.22	0.34 ± 0.09	0.33 ± 0.08
A-loop...C-lobe				
Y823...R796	1.16 ± 0.11	0.57 ± 0.27	0	0
Y823...Y846	0	0	0.03 ± 0.02	0.1 ± 0.05
C-loop				
D792...H790	1.51 ± 0.02	1.35 ± 0.17	1.35 ± 0.13	1.11 ± 0.15
H790...N797	0	0	0.06 ± 0.06	0.18 ± 0.18
D792...N797	2.01 ± 0.08	1.71 ± 0.09	1.39 ± 0.22	1.29 ± 0.26
C-loop...A-loop				
D792...Y823	1.3 ± 0.1	0	0	0
C-loop...C-lobe				
N797...C809	0.34 ± 0.04	0.33 ± 0.16	0.45 ± 0.11	0.61 ± 0.15
C-lobe				
L799...K807	2 ± 0	2 ± 0	2 ± 0	2 ± 0

Table 3.3: *Average H-bond number between residues in CHARMM36 force field simulations of KIT WT and MU. ± indicates standard error across independent simulations.*

	KIT-I*^{TP1}	KIT-I*^{TP2}	KIT-A*^{TP1}	KIT-A*^{TP2}
JMR...C-lobe				
V560...N787	1.1 ± 0.05	1.1 ± 0.02	0.02 ± 0.02	0.05 ± 0.04
K558...I789	1.65 ± 0.13	1.34 ± 0.13	0	0
Y568...F848	0.59 ± 0.028	0.74 ± 0.27	0	0
Y570...Y846	0.32 ± 0.06	0.06 ± 0.01	0	0
E640...F811	0	0	0.57 ± 0.13	0.84 ± 0.08
JMS				
V569...E561	1.11 ± 0.05	1.17 ± 0.14	0	0
A-loop				
V824...L831	1.94 ± 0.06	1.64 ± 0.22	0	0
R815...Y823	1.14 ± 0.35	0.89 ± 0.46	0.31 ± 0.17	2.09 ± 0.66
D820...N822	0.79 ± 0.14	0.36 ± 0.09	0.49 ± 0.24	0.23 ± 0.08
A-loop...C-lobe				
Y823...R796	2.11 ± 0.54	2.87 ± 0.05	0	0
Y823...Y846	0	0	0.83 ± 0.45	0.08 ± 0.05
C-loop				
D792...H790	1.65 ± 0.1	1.32 ± 0.1	1.26 ± 0.15	1.53 ± 0.1
H790...N797	0	0	0.2 ± 0.11	0.03 ± 0.02
D792...N797	1.72 ± 0.08	1.73 ± 0.05	0.95 ± 0.2	1.49 ± 0.14
C-loop...A-loop				
D792...Y823	0.57 ± 0.25	0	0	0
C-loop...C-lobe				
N797...C809	0.52 ± 0.12	0.2 ± 0.08	0.59 ± 0.35	0.4 ± 0.17
C-lobe				
L799...K807	2 ± 0	2 ± 0	1.99 ± 0.01	1.99 ± 0.01

Table 3.4: *Average H-bond number between residues in CHARMM36 force field simulations of TP1 and TP2. ± indicates standard error across independent simulations.*

beta-turn motif [194]. We also observe different χ angle distributions of D810 and F811 of the DFG motif between the KIT-I^{WT} and KIT-I^{MU} (Figure 3.12).

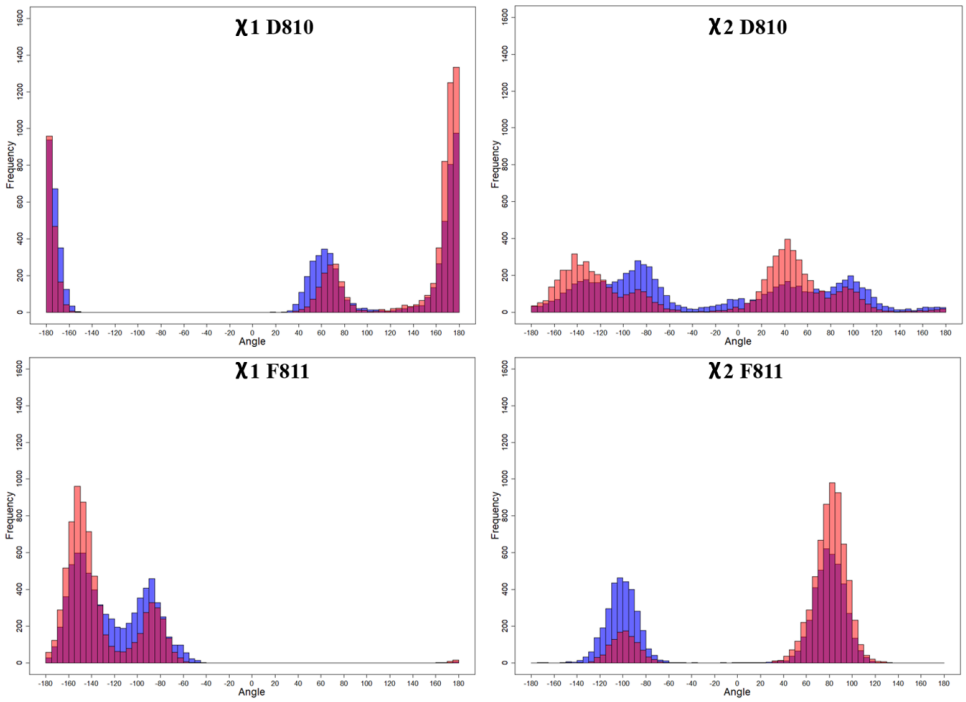

Figure 3.12: *Distribution of D810 and F811 sidechain χ angles between KIT-I^{WT} (blue) and KIT-I^{MU} (red).*

In the KIT active state, the residues D792, H790, and N797 of the C-loop (catalytic loop) form a stable local H-bond interaction network along the MD trajectories. Further, the H-bond analysis shows stabilization of JMR, A-loop, and C-loop, suggesting that the mutation Y823D and the phosphorylation of Y823 cause stabilization of the active state. In particular, the H-bonds R815...Y(D)823 and Y(D)823...Y846 are observed more frequently in all simulations for the mutated and phosphorylated KIT in the active state. These H-bonds are located either in the A-loop or connect the A-loop and the C-lobe, probably stabilizing their mutual orientation. On the contrary, in the mutated and phosphorylated protein in the inactive state, the interactions of JMR with the KD are preserved, while the A-loop is displaced and destabilized.

3.2.2.5 CHANGES IN THE CORRELATED MOTIONS OF AMINO ACID RESIDUES

The correlation between the motions of all pairs of residues emerges from the various ways they interact. Here, we investigated how the mutation Y823D alters the concerted movement between residues, affecting the allosteric communication between distant regulatory regions of KIT. To do so, we computed the mutual information (MI) between trajectories of individual residues. MI reflects the degree to which two random variables are linked; high MI indicates a low uncertainty in one random variable given the information about the other. Interestingly, in active and inactive states, most correlations are between distant residues (Figures 3.13A, 3.13D, and 3.14). We can observe that only a few residues in the mutated protein in the active state show changes in the correlated motion and that almost all the residues involved in the interactions are between 8 and 21 Åapart from one another. Additionally, in the inactive state, we see that the changes in the pattern of correlated motions are much more substantial in the mutant protein than in the WT.

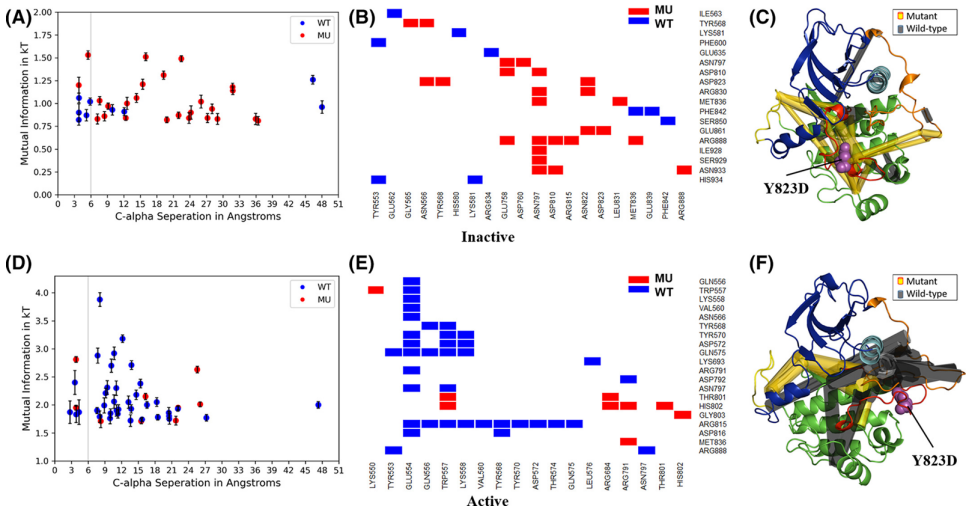

Figure 3.13: *(A, D) Distance between correlated residues (MI > 0.8 kT in the inactive state and > 1.7 kT in the active state). A grey line separates local (<6 Å) and long-range interactions. Standard deviation from the bootstrapped dataset is plotted as the error bars on top of the MI values from the actual (not bootstrapped) simulations. (B, E) Correlations with MI greater than 0.8 kT for the inactive state and MI greater than 1.7 kT for the active state are only shown. (C, F) Cylinders connecting residues represent differences in MI between those residues in MU (yellow) and WT (grey), with the width of the cylinder proportional to the MI. Mutation site Y823D is represented as spheres in magenta.*

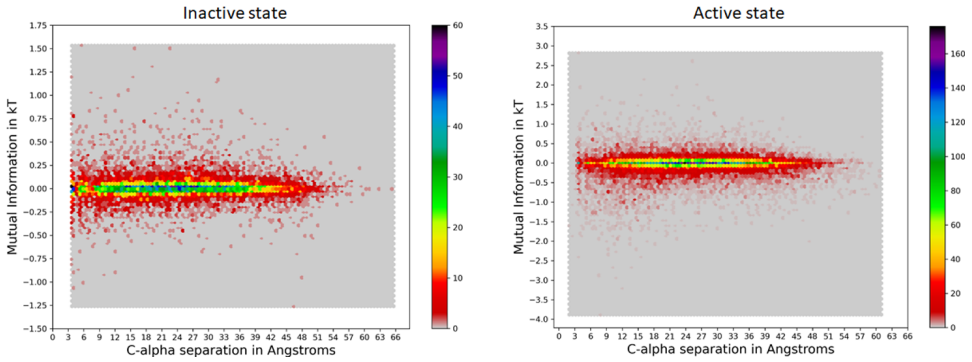

Figure 3.14: *The overall distribution of mutual information (MI) of WT (above the X axis) and MU (below the X axis) of inactive and active states without a threshold.*

Further, analysis of the correlated motion of residues in the inactive and active states revealed that specific residues act as hubs connected to several other residues (Figure 3.13B, E). However, the residues acting as hubs are not the same for active and inactive states, indicating that the correlated motions are different in different conformations of the same protein and that mutation Y823D affects these conformations differently. In the inactive state, the residues N797 and R888 act as hubs interacting with several key residues (D810, R815, R830, M836) of the protein (Figure 3.13B), suggesting that any change in the motion of these residues will affect the correlated movement of the other residues. Similarly, in the active state, the residues E554 and the highly conserved residue R815 act as hubs. Overall, we see that more residue pairs are involved in correlated motions in the WT than in the mutant protein (Figure 3.13E). Among the residues whose movement was most affected by the mutation in the inactive state are the residues of the inhibitor binding site and the conserved C-loop residues (residues 568, 797, 810, and 815), P+1 loop (the loop immediately following the A-loop, residues 830, and 836), critical residue of the DFG motif (residue 810) and F-helix (residue 861).

In KIT-I^{WT}, the signal from the A-loop to the JMR is propagated through the C-loop residues, where Y823 acts as an intermediate residue [194]. From MI data, we can observe how a mutation in this position has changed the communication between distant regulatory regions in the protein. Thus, the modification in the local interaction network of the C-loop that we observed in the H-bond analysis is accompanied by a reduction in the efficient communication to the distant KIT regulatory elements. In the active state of KIT, we notice a different pattern, where

none of the correlated motions below the threshold (1.7 kT) involving the key residues were affected by the mutation (Figure 3.13).

3.2.2.6 PARTIAL LEAST-SQUARES REGRESSION

The functional mode analysis based on the partial least-squares (PLS) regression method aids in differentiating the significant collective motions between the WT and MU proteins. We used the coordinates of the backbone atoms as input, which proved to have the most predictive power when creating the statistical models for the inactive state (data for other input types not shown). For the active state, none of the input features could provide a satisfactory model that could be used to differentiate the collective motions between the WT and mutant proteins. Therefore, only the results for the inactive state of the protein are discussed.

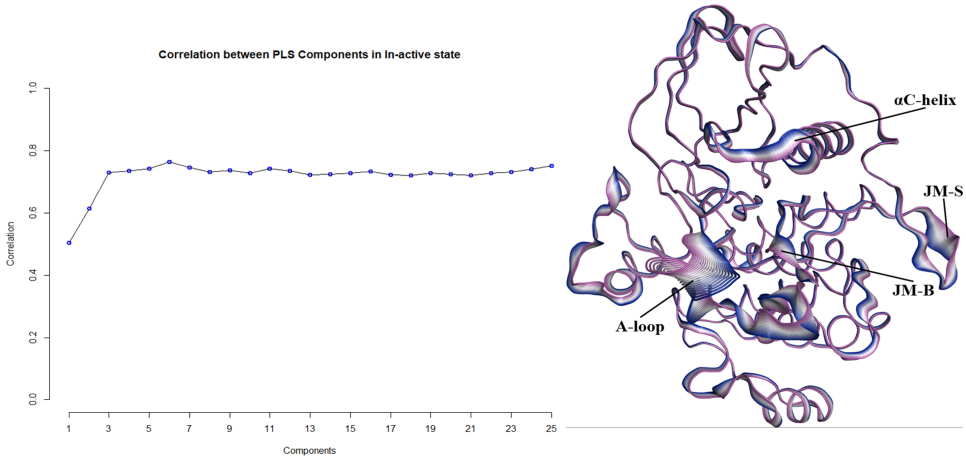

Figure 3.15: *Left: Cross validation: Correlation between predicted and true label for the backbone of trajectory. Right: Interpolation between the extremes of the PLS models. Blue-to-magenta bands correspond to the interpolation along the mode which relates to the true label of simulation, wild-type and mutant, to the underlying differences in the protein motions.*

We used the four-fold cross-validation (CV) technique to obtain the statistical models. We measured the prediction quality through the Pearson correlation score between the true and predicted labels for a given input feature. We chose models composed of 3 PLS components to study the changes in the collective motions of the protein, which was sufficient in the CV scenario to achieve a correlation of 0.75 (Figure 3.15). Mapping the PLS results onto the backbone of the inactive state of KIT (Figure 3.15) showed the mutation-induced changes in the collective motions in the JM-B, JM-S, A-loop, and αC-helix. In agreement with

this, we observe that the same regions undergo large shifts in the PCA analysis.

3.2.3 DISCUSSION

These results from this case study represent an advancement by applying MD simulations to investigate the effects of Y823 phosphorylation on the KIT protein alongside the previously studied mutation Y823D . While the mutation Y823D , which was the focus of my master's thesis, has been associated with resistance, among others, to imatinib and sunitinib [228, 231–233] and is a secondary mutation in gastrointestinal stromal tumors [228, 233, 234], this case study part of my PhD introduces new insights into the role of Y823 phosphorylation, a modification known to stabilize the active conformation of the A-loop, most probably by strong electrostatic interactions of the phosphate group [224, 225]. One key challenge in this study was parameterizing the phosphorylated residues in the KIT protein models. Several attempts using Amber99SB*-ILDN force fields were unsuccessful, as they did not adequately support the complex interactions and charge distributions associated with these residues. Consequently, we utilized the CHARMM36 force fields, which allowed us to successfully parameterize the phosphorylated residues in the KIT protein models by incorporating relevant parameters based on the literature. This allowed us to study the stabilizing effects of Y823 phosphorylation on the KIT protein, offering new insights into its dynamic behavior and functional implications that were not previously accessible.

Our analysis reveals that both the Y823D mutation and the phosphorylation of Y823 stabilize the protein's active conformation, particularly by strengthening the H-bonds, such as the one between R815 and Y823. This stabilization is consistent with previous findings [154, 156, 224, 225], and our study extends this understanding by showing how phosphorylation affects protein dynamics. We observe that in the active state, both the mutation and phosphorylation improve the stability of the A-loop, which is crucial for maintaining the active conformation. In line with our observations, mutations in the A-loop are known to disrupt the inactive conformation by introducing charged side chains into the pocket [154, 231, 233].

In our comparative analysis, the different dynamics observed in the active and inactive conformations of the wild-type, mutant, and phosphorylated KIT proteins demonstrate distinct behaviors in the same structural regions: the active site residues and key residues in the A- and C-loops (Figure 3.16). These dynamics are notably different for the inactive state of KIT, whereas changes are less pronounced in the active state. Notably, the dynamics of the mutant protein consistently resemble those of the kinase phosphorylated at the mutation site Y823. This similarity is likely due to the negatively charged aspartate at Y823 effectively mimicking the effects of phosphorylation at this site.

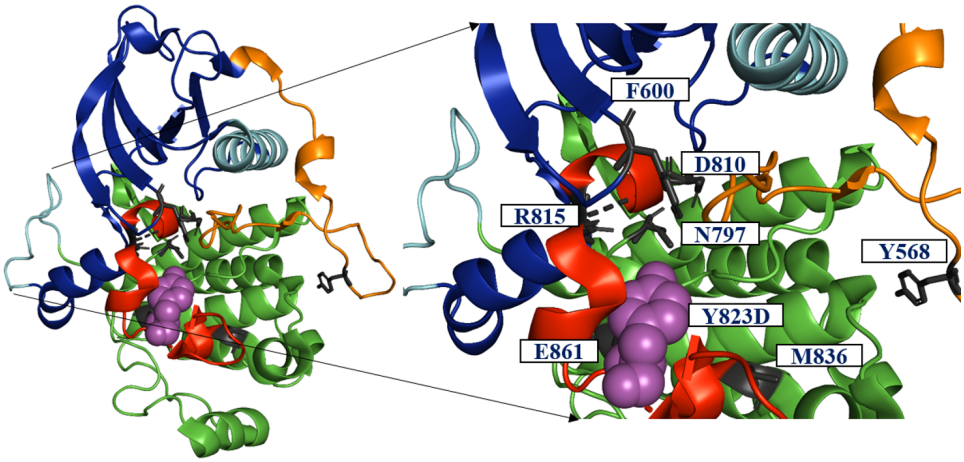

Figure 3.16: *Active site residues were mapped on the KIT inactive state protein structure.*

The biological significance of these findings aligns with the established literature [194, 235–237]. In particular, the C-loop is highly conserved in structure and sequence among all kinases [235]. Another structural element, the F-helix, whose motion was also affected in our analysis, was shown to play an essential role in protein kinase active structures, as it is considered the central hub connecting key areas such as the substrate binding residues and the C-loop. Also, the regulatory and catalytic spines are located at the N and C termini of the F-helix [194, 236, 237]. This puts the perturbations observed through RMSF analysis in the JM-B, JM-S, A-loop, and the residues preceding the αC-helix in the inactive state and the local destabilization of the A-loop through loss of H-bonds into a proper functional context. Most importantly, we observe the loss of allosteric communication, as evident from the changes in the hydrogen bond network reported above, between distant regulatory

domains, namely JMR and the A-loop, which may be detrimental to kinase regulation. As observed from this analysis, the mutation Y823D and the phosphorylation of Y823 stabilize the active conformation of the protein by contributing to the formation of more stable H-bonds in the A- and C-loops.

The concerted alterations that we observe in all simulations of the inactive state may indicate mechanisms that affect the binding of drugs. Multiple lines of evidence presented here suggest that the changes in the structural conformation of the inactive state protein correspond to its destabilization upon mutation and phosphorylation and a shift of the dynamic equilibrium away from the inactive state, which is a primary target for many drugs. For example, imatinib specifically binds to the inactive state of KIT [156, 238]. Thus, these changes in the key regions of the protein's inactive state may alter its binding and/or sensitivity to it and other drugs that explicitly target the inactive state of KIT.

In summary, we demonstrated that while the phosphorylation of Y823 and mutation Y823D stabilize the KIT protein's active state, they do so through distinct mechanisms. The mutation Y823D mimics the effects of phosphorylation by introducing a negative charge at Y823, destabilizing the inactive conformation and shifting the conformational equilibrium towards the active state. This shift is accompanied by changes in the correlated motion of amino acids, particularly between the mutation site and the active site residues, and other key residues such as D792, R815, and Y823 [11, 156, 219] in the A-loop and catalytic loop, which are involved in drug binding and allosteric communication. This change will likely activate downstream signaling cascades and enhance the expression of anti-apoptotic, cell proliferation, and growth expression genes. Phosphorylation at Y823, on the other hand, enhances the stability of the active state by forming more robust H-bonds and strengthening interactions within the A-loop. Our comparative analysis suggests that while both the mutation and phosphorylation promote the active conformation, the mutation more directly disrupts the inactive state by compromising the communication between vital regulatory regions of the protein and inducing local destabilization due to the loss and reduction of key H-bonds. Notably, Y823 in KIT is homologous to Y393 in the Abl kinase, where phosphorylation of this residue is known to destabilize the inactive conformation of the A-loop [224]. In our simulations, we confirm a

similar trend for the mutation Y823D in KIT by conducting simulations with a phosphorylated tyrosine at this position. Thus, we suggest that an analogous resistance mechanism may have evolved in a homologous kinase, leading to similar pathologic consequences.

3.3 EPISTATIC INTERACTIONS AND PERSISTENCE OF NS3-Q80K IN HCV: A MOLECULAR DYNAMICS PERSPECTIVE ON DRUG RESISTANCE

Here, we investigated how epistatic interactions, particularly between substitutions A91S, A91T, and S174N, drive the persistence of the NS3-Q80K mutation within the nonstructural protein NS3 of HCV. NS3, encoded by the HCV genome, is a multifunctional protein crucial for viral replication and assembly, functioning as both a protease and an RNA helicase. Understanding this persistence is vital for HCV pathophysiology, as the NS3-Q80K mutation correlates with diminished treatment response to direct-acting antiviral agents (DAAs). This study demonstrates how these epistatic interactions compensate for protein folding instability, thereby allowing the NS3-Q80K mutation to persist.

3.3.1 BACKGROUND: AN OVERVIEW OF HEPATITIS C VIRUS

HCV is a major bloodborne pathogen that causes chronic hepatitis, cirrhosis, and hepatocellular carcinoma (HCC) and is considered a significant public health issue [239]. Six known genotypes (HCV-1 to HCV-6) and more than 50 subtypes (e.g., 1a, 1b, 2a) have subsequently been identified, with different geographical and virulence patterns and responses to conventional therapy [240]. At present, the World Health Organization (WHO) estimates that about 50 million people are chronically infected with HCV, and about 1.0 million new infections occur every year.

Hepatitis viruses A (infectious hepatitis virus) and B (serum hepatitis virus) were only recognized from 1975 [241]. During tests applied to sera from patients who acquired these viruses after transfusion, it was found that none of the cases were caused by hepatitis A, resulting in the new terminology non-A and non-B hepatitis [242]. Through classical virological methods, it was characterized that transmissible agents cause these. In 1989, the genome was cloned for the first time, and diagnostic tests

were developed [243]. Soon after the non-A and non-B were renamed as hepatitis C virus, it was identified that the biological and molecular characteristics are closely related to the *Flaviviridae* virus family [241].

3.3.1.1 HCV GENOME

Hepatitis C virus (HCV) is an enveloped, single-stranded, positive-sense RNA virus of approximately 50 nm in diameter of the *Hepacivirus* genus of the family *Flaviviridae* [239, 244, 245]. Its genome, approximately 9.6 kilobases in length, encodes a single open reading frame (ORF) flanked by untranslated regions (UTRs). Within this ORF, a polyprotein precursor of about 3,000 amino acids is synthesized, which is processed into structural and non-structural proteins essential for viral replication and assembly. Lipoproteins in HCV facilitate the entry of HCV into hepatocytes by endocytosis [241]. The process involves several steps involving viral proteins and host factors. Upon entry into the host cell, the viral RNA is translated into a polyprotein, which is then cleaved by viral and host proteases into nonstructural and structural proteins, respectively.

The HCV RNA (Figure 3.17) contains an internal ribosome entry site (IRES) within its 5′ UTR, which facilitates cap-independent translation initiation, unlike the others in the *Flaviridae* genera, which are cap-dependent. This allows the viral RNA to be efficiently translated into protein despite lacking a 5′ cap structure typical of host mRNA. Similarly, the flavivirus 3′ UTR is highly structured. In contrast, the HCV 3′ UTR is short and less structured, holds a replication element comprising a variable RNA stem-loop, and contains a polyuridine/polypyrimidine tract of varying length.

The structural proteins (Figure 3.17) include the core protein (C) and the envelope glycoproteins E1 and E2, essential for virion assembly and entry into host cells. The nonstructural proteins (Figure 3.17), including NS2, NS3, NS4A, NS4B, NS5A, and NS5B, are involved in viral replication, assembly, and evasion of host immune responses. The nonstructural proteins play critical roles in HCV replication, forming complexes and modifying host cell processes to create a favorable environment for viral replication. For example, NS3 functions as a serine protease and helicase, NS4A acts as a cofactor for NS3 protease activity, and NS4B, an integral membrane protein, plays a crucial role in assembling the membranous

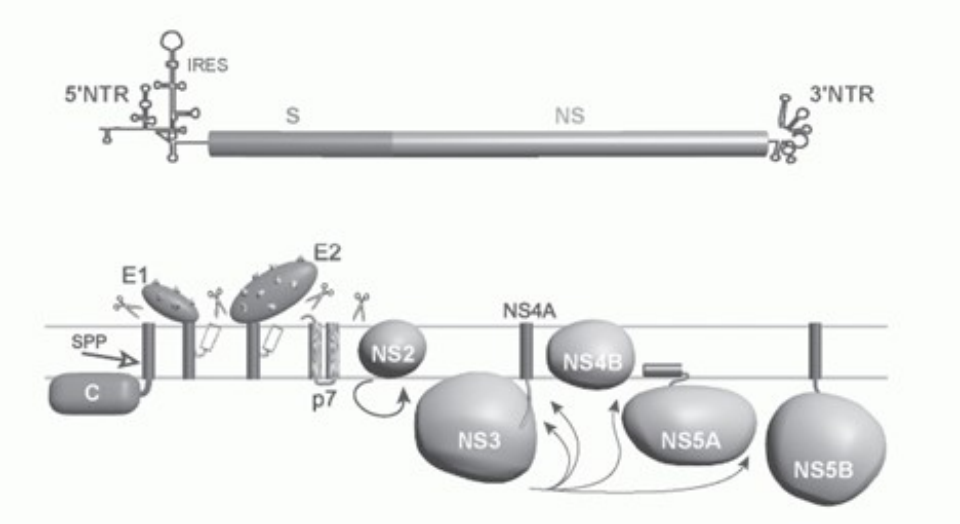

Figure 3.17: *(Source: (Chevaliez et al., 2006 [240])) HCV genome organization.*

web, which serves as the organelle utilized for RNA replication. NS5A is a phosphoprotein involved in viral RNA replication and modulation of host cell signaling pathways and virus assembly, and NS5B serves as the RNA-dependent RNA polymerase responsible for viral RNA synthesis.

3.3.1.2 HCV DRUG RESISTANCE

The emergence of resistance mutants poses a significant challenge to developing effective antiviral treatment regimens for HCV. Approximately 50 million people are chronically infected with HCV and are at risk of developing cirrhosis, HCC, liver disease, etc., with an increasing number of mortality cases. Due to the error-prone nature of viral RNA replication and the lack of proofreading by the NS5B polymerase, the diversity of HCV genotypes and subtypes contributes to variations in the clinical outcomes of therapies. Resistance mutants have been observed to grow, impacting the effectiveness of antiviral drugs targeting the key viral proteins [246].

For the past decade, conventional therapy for patients with chronic HCV infection consisted of administering pegylated interferon alfa (PEG-IFN) alongside ribavirin (RBV). However, this treatment only achieved sustained virologic response rates, defined as negative HCV-RNA, in approximately 40-50% of patients [247]. Advancements in the field have led to the understanding of the viral life cycle, and the crystal structure of the essential viral proteins has paved the way for creating treatment

protocols comprising interferon-free combinations of direct-acting antivirals (DAAs) administered orally [246–248]. These regimens significantly vary from previous therapies, providing improved efficacy and tolerability [249]. DAAs specifically target the key viral proteins involved in viral replication and assembly. Thus disrupting its lifecycle and inhibiting its ability to replicate and spread. It offers the potential to intervene at various stages of the viral lifecycle. Hence, the primary targets of these DAAs are the nonstructural proteins of HCV, particularly the NS3/4A protease, NS5A, and NS5B polymerase. By inhibiting the function of these proteins, DAAs prevent viral replication and assembly [248].

3.3.1.3 NONSTRUCTURAL PROTEIN (NS3)

The multifunctional NS3, a 631 amino acids long protein encoded by the HCV genome, is pivotal in viral replication and assembly. NS3 comprises two distinct functional domains, which are separated by a deep cleft that harbors the active site (Figure 3.18): the first domain, in the N-terminal one-third of NS3, has a chymotrypsin-like serine protease necessary for cleaving at four specific sites within the nonstructural region of the HCV polyprotein. The remaining two-thirds (the C-terminal) of NS3 contain a helicase and a nucleic acid-stimulated nucleoside triphosphatase (NTPase) [155, 250, 251]. NS3 forms a non-covalent complex with NS4A, a short protein of 54 amino acids, activating the NS3 protease [155, 252]. Without NS4A, the 30 N-terminal residues of NS3 lack a defined structure and extend outward from the protein. However, when an NS4A peptide is present, the overall structure of the C-terminal domain remains unchanged. In contrast, the N-terminal domain undergoes a structural change, forming eight-stranded β-barrel strands [250, 251, 253]. This β-barrel includes one strand contributed by NS4A and tightly integrates with the N-terminal domain of NS3 through hydrogen bonds, making it an essential part of the NS3 protease. This structural arrangement enhances the stability and functionality of the NS3-4A complex, facilitating its role in HCV replication and viral protein processing. NS3's protease activity is crucial for cleaving the viral polyprotein into functional components necessary for viral replication, while its RNA helicase activity facilitates the unwinding of viral RNA structures during replication. Additionally, NS3 interacts with NS5A, another essential nonstructural protein, regulating viral RNA replication and host cell

signaling pathways. These interactions highlight how NS3 is crucial in controlling different phases of the HCV lifecycle, including viral entry, replication, and assembly inside host cells [251].

The HCV protease NS3/4A can cleave the cellular targets involved in innate immunity and is essential for viral infectivity. Thus, it is a promising target for antivirals [239]. However, resistance-associated substitutions (RASs) can hinder the effectiveness of DAA therapies for HCV. Even before treatment, some patients may naturally harbor RASs, impacting treatment outcomes. For instance, the presence of RAS Q80K, a naturally occurring NS3 RAS, can reduce the effectiveness of simeprevir. Therefore, guidelines suggest screening for Q80K in HCV genotype 1a patients before treatment to ensure optimal therapy selection [254–257]. RASs, such as the Q80K, can significantly impact and alter the structural and functional properties of NS3, rendering it less susceptible to inhibition by DAAs. Q80K mutation is known to have a negative impact on several combinations of antivirals in clinical studies. Consequently, drug-resistant variants of HCV emerge, hindering the efficacy of antiviral therapies and posing challenges in managing HCV infections. Understanding the structural and functional changes due to such mutations and the effects of compensating mutations is imperative for devising effective strategies to combat drug resistance and enhance the efficacy of antiviral treatments.

The S174N mutation is a significant phylogenetic marker [258, 259], facilitating clade differentiation by being present exclusively in clade 1 sample and absent in clade 2. This mutation, along with A91S/T, forms part of the epistatic interactions identified in the evolutionary trajectory of the Q80K mutation within the NS3 protease [258, 259]. These interactions potentially compensate for structural or functional alterations induced by Q80K. Epistasis arises when the collective impact of multiple mutations deviates from the simple addition of their individual effects, suggesting interactions at the molecular level that influence protein function. Investigating such complex interactions sheds light on essential intramolecular networks driving protein evolution and function. The NS3 protease, a key target against HCV, is conserved across *Flaviviridae* and is targeted by new drugs for emerging pathogens [240, 251, 260]. RASs in NS3 often incur fitness costs, primarily reducing RNA replication capacity and, in some cases, impairing virus production [261].

In this study, we explore the epistatic interactions that facilitate the stabilization of the NS3 protease protein fold in Q80K variants, restoring protein stability comparable to that of the wild-type (WT).

3.3.2 RESULTS

Along with our experimental collaborators, we explored how interactions between different sites in the NS3 protease structure, known as epistasis, may explain the persistence of NS3-Q80K in genotype 1a HCV infection. Mutant patterns were selected based on a study [258]. We analyzed genotype 1a-infected patients with NS3-Q80K, examining epistatic secondary substitutions at residue 91 and/or 174 based on sequence information from DAA-experienced patients from the Frankfurt Resistance Database and sequence information from DAA-naïve patients from the European HCV database [262].

I will first present the bioinformatics analysis I carried out and partially derived from our collaborative publication. Then, I will summarize the experimental results for completeness and include relevant figures from the publication in Chapter 3.3.2.2.

3.3.2.1 MOLECULAR DYNAMICS SIMULATIONS ANALYSIS

We analyzed the protein dynamics and inter-residue contacts that could explain the protease fold stabilization in NS3-Q80K harboring epistatic secondary substitutions. The Q80 residue resides in the protease–helicase domain interface of NS3 close to the protease active site (Figure 3.18). The epistatic site S174 resides near Q80 (approximately 3.9 Ådistance), exhibiting hydrogen bonds between both residues. In contrast, the second epistatic site, A91, is more than 20 Ådistant from residue Q80 (Figure 3.18). For simulations, we created protease mutant structures corresponding to NS3-Q80K with or without epistatic amino acid substitutions (A91S, A91T, S174N).

The Root Mean Square Deviations (RMSDs) were calculated for the backbone atoms of the NS3 protease variants over the concatenated trajectories, excluding the first 10 ns from each of the four 100 ns MD simulations. The backbone RMSD profiles (Figure 3.19) show that they

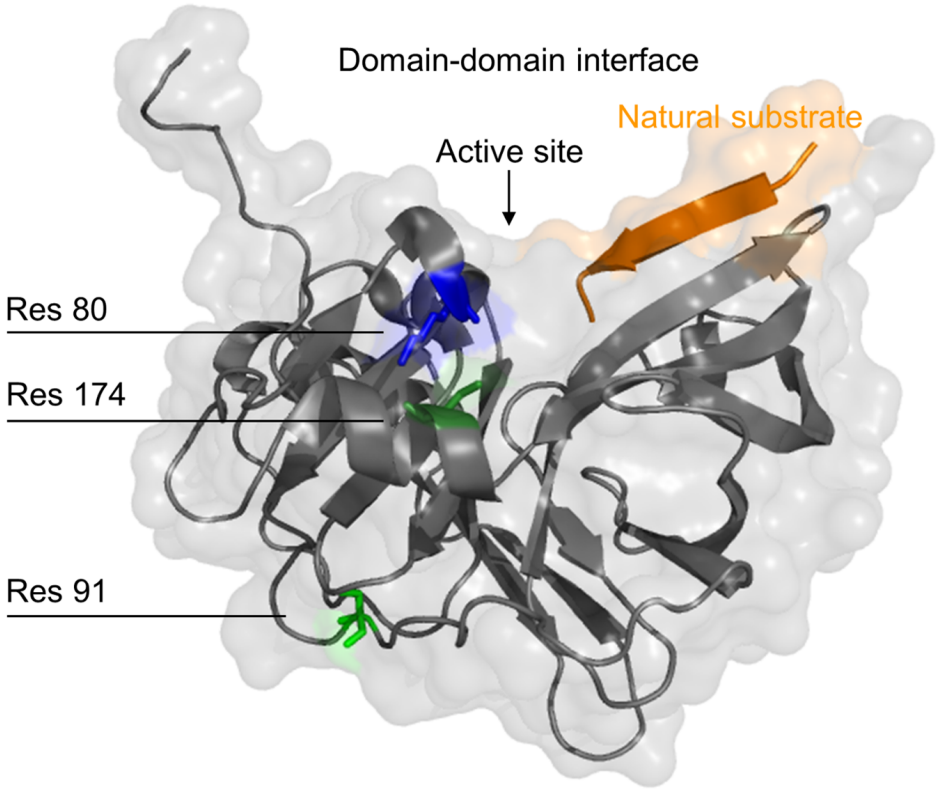

Figure 3.18: *Protease structure with co-crystallized natural peptide substrate (in orange) (PDB ID: 3M5O [263]); the protein backbone is shown as a ribbon model with a transparent surface depiction. Residues of interest are highlighted as stick models and colored as follows: residue* 80*, blue; residue* 91*, light green; and residue* 174*, dark green.*

conform closely to their initial crystallographic structures, and the mutant variants exhibit minimal deviations.

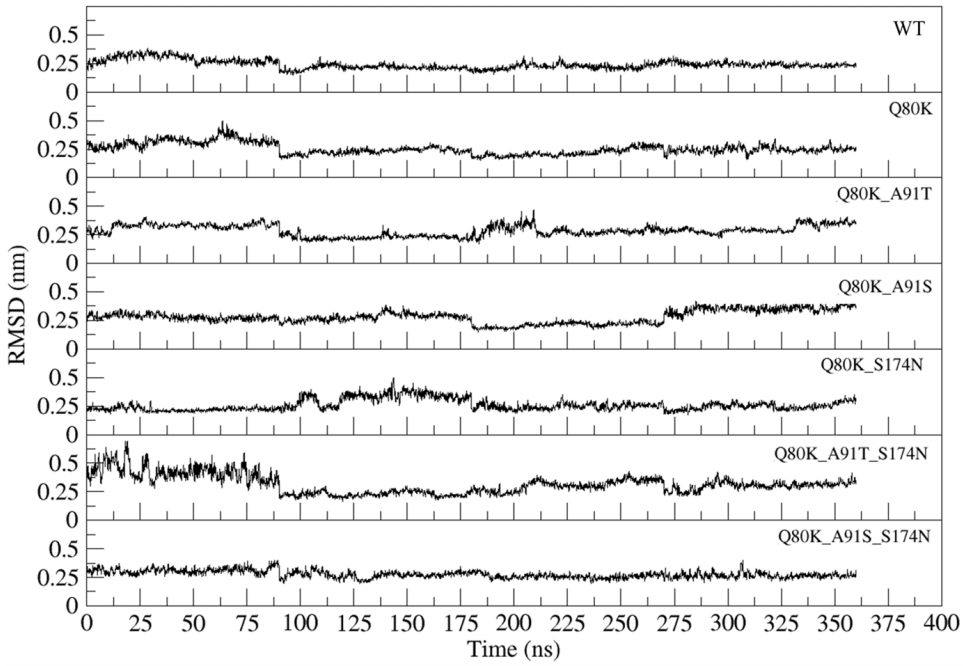

Figure 3.19: *The RMSD values were calculated for the backbone atoms of the concatenated trajectories, combining four replicate simulations for each variant. The data compare the RMSF of the WT NS3 protease structure (PDB ID:* 1CU1 *[164]) with the mutant structures carrying the NS3-Q80K mutation along with the epistatic amino acid substitutions: A91S, A91T, and S174N.*

The Root Mean Square Fluctuations (RMSF) (Figures 3.20 and 3.21) show that the NS3-Q80K mutation leads to increased flexibility and potential destabilization of the protein structure, as observed by increased RMSF values upon Q80K mutation. However, introducing secondary substitutions, such as S174N and/or A91S/T, alleviates this destabilization (Figures 3.20 and 3.21). These epistatic substitutions reduce flexibility and enhance protein stability, as reflected in the RMSF plots. These findings highlight the significant role of epistatic substitutions in counteracting the destabilizing effects of the NS3-Q80K mutation.

The modeling results are consistent with the protein denaturation curves (Figure 3.23) observed for NS3-Q80K and epistatic secondary substitutions (Figures 3.20 and 3.21), indirectly supporting our hypothesis on the differential effects of mutations on the bilobal structure of the NS3 protease. Mutants containing S174N or A91T tended to stabilize the fold, as observed from the RMSF values for each trajectory (Figure 3.20). We have quantified this observation by extracting 181 frames

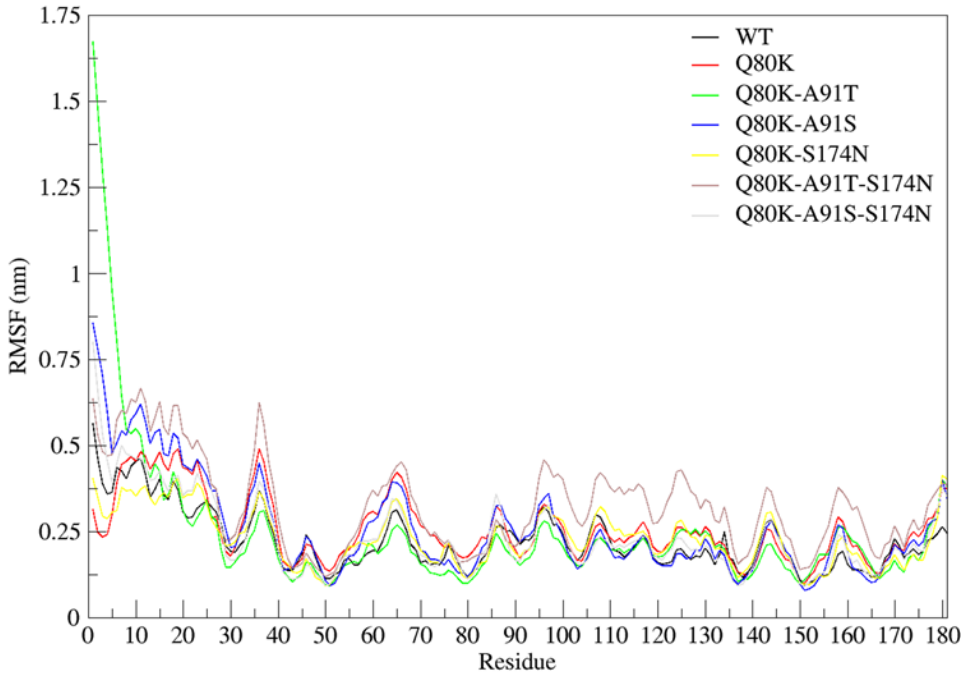

Figure 3.20: *The RMSF values computed on the backbone atoms of the concatenated trajectory of MD simulations. The data compare the RMSF of the WT NS3 protease structure (PDB ID:* 1CU1*) with the mutant structures carrying the NS3-Q80K mutation along with the epistatic amino acid substitutions: A91S, A91T, and S174N.*

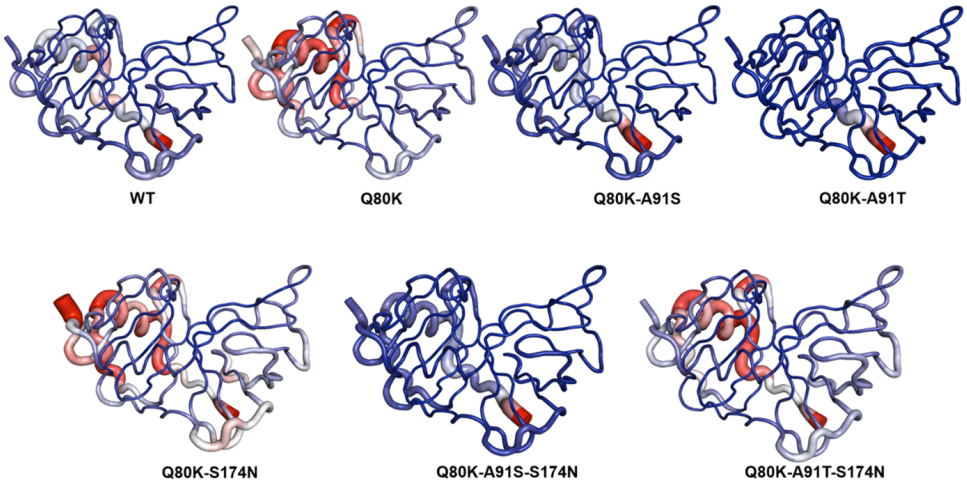

Figure 3.21: *Protease protein flexibility averaged over time for different NS3-Q80K mutant patterns. NS3 protease WT (PDB ID:* 1CU1*) and NS3-Q80K mutants are represented as tubes. The highly flexible residues are shown in red, the highly stable residues in blue, and the intermediate flexible residues in white. The size of the tubes is proportional to the observed RMSF values in those regions (Figure 3.20), respectively.*

from the concatenated MD trajectory and computing the number of contacts for each amino acid substitution at positions 80, 91, and 174 (Figure 3.22). We observe that while the number of contacts of individual positions fluctuates, notably when the number of contacts of position 80 decreases upon Q80K mutation, the total number of contacts correlated well with the RMSF values from the MD trajectories and with the stability measurements (Figure 3.24), increasing with substitutions at Q80K-A91T, Q80K-A91T-S174N, and Q80K-A91S-S174N. Notably, new contacts and hydrogen bonds formed by the mutated amino acid at position 91 contribute most to this trend. This finding agrees with previous evidence that long-range contacts generally make a more significant thermodynamic impact than contacts near in sequence and structure [264].

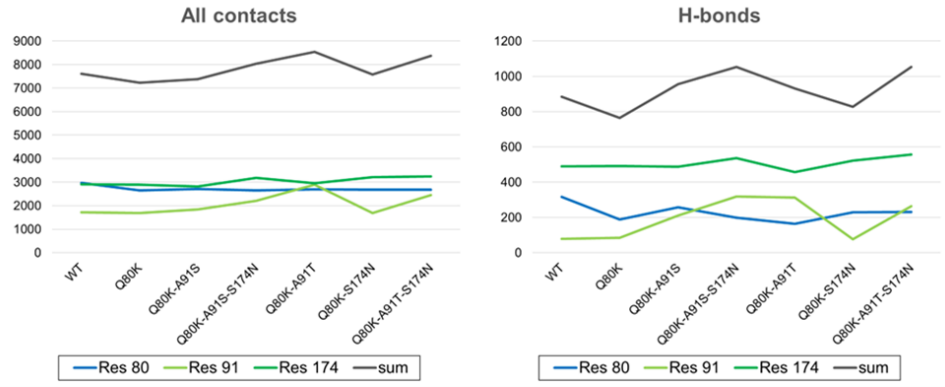

Figure 3.22: *The number of contacts quantified from concatenated MD trajectories. The total number of contacts (left) and the number of hydrogen bonds (right) are shown in equally spaced* 181 *frames from the MD trajectories.*

3.3.2.2 A SUMMARY OF THE EXPERIMENTAL DATA ANALYSIS

To examine how the Q80K mutation in NS3 and its epistatic amino acid substitutions S174N and A91S/T influence protein folding, our experimental collaborators expressed and purified both the WT and mutant NS3-4A proteins. Using a thermal shift assay, which measures protein denaturation at increasing temperatures via fluorescence, they assessed if NS3-Q80K with or without S174N and/or A91S/T affected protease stability by measuring protein denaturation at increasing temperatures. Results (Figures 3.23, 3.24) showed NS3-4A WT remained stable up to a half-maximal denaturation (T_m) of 50.6°C, while NS3-Q80K signifi-

cantly destabilized at a T_m of 44.1°C ($p < 0.0001$). All epistatic mutations raised denaturation temperatures and stabilized the protease fold of the respective expressed NS3-Q80K mutant proteins, except Q80K-A91S, which did not reach WT protein stability levels (Figure 3.24). The double mutants Q80K-A91T (T_m at 49.6°C, $p < 0.0001$) and Q80K-S174N (T_m at 48.4°C, $p < 0.0001$), along with triple mutants Q80K-A91S-S174N (T_m at 48.6°C, $p = 0.018$) and Q80K-A91T-S174N (T_m at 50.1°C, $p < 0.0001$), showed stability comparable to the WT protease (Figure 3.24, Table 3.5). Deviations from expected denaturation curves suggest Q80K affects protease subdomains and communication between them, possibly influencing catalytic activity at the active site. The Q80K mutation presented a distinct shoulder during denaturation (Figure 3.23A), indicating differential effects on the bilobal structure of the protease and causing separate denaturation of the subdomain with the mutation upon heating. Additionally, epistatic secondary substitutions that compensate for Q80K also displayed shoulders and non-parallel denaturation curves relative to WT protein (Figure 3.23B-F), further supporting the disruption of inter-lobal communication.

Variant	K_M **(μM)**	k_{cat} **(/min)**	T_m **(°C)**
WT (Q80)	2.71 ± 0.31	5.46 ± 0.26	50.6 ± 0.15
Q80K	1.44 ± 0.24	1.8 ± 0.1	44.1 ± 0.29
Q80K-A91S	0.56 ± 0.06	1.56 ± 0.06	45.0 ± 0.19
Q80K-A91T	0.48 ± 0.13	0.4 ± 0.03	49.6 ± 0.16
Q80K-S174N	0.14 ± 0.03	0.48 ± 0.02	48.4 ± 0.37
Q80K-A91S-S174N	8.16 ± 4.62	0.13 ± 0.04	48.6 ± 1.33
Q80K-A91T-S174N	0.67 ± 0.09	2.41 ± 0.11	50.1 ± 0.21

Table 3.5: *Enzyme kinetic constants and protease protein T_ms for the WT protease and NS3-Q80K variants with epistatic amino acid substitutions. FRET-based protease and thermal shift assay obtained Kinetic constants and melting temperatures, respectively. The data shown represent the mean ± SD from at least three independent experiments.*

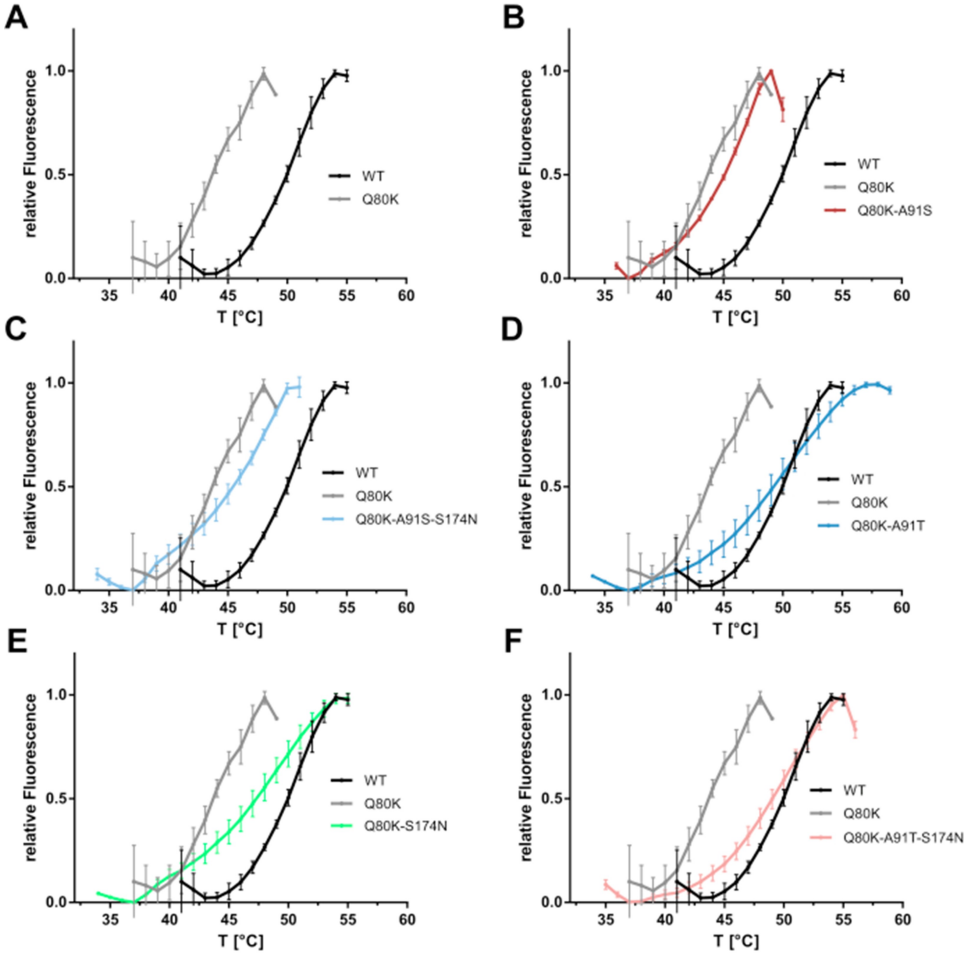

Figure 3.23: *Thermal shift of the NS3-4A protease harboring NS3-Q80K and epistatic amino acid substitutions. Data from real-time thermal stability assay using Sypro Orange, a temperature-stable fluorophore that exhibits enhanced fluorescence upon interacting with unfolded proteins. (A) The thermal stability of the WT protease, the NS3-Q80K mutant, and (B-F) mutants harboring epistatic secondary substitutions were assessed under increasing incubation temperatures—graphs showing the impact of mutations on protease protein unfolding patterns characterized by fluorescence emission curves.*

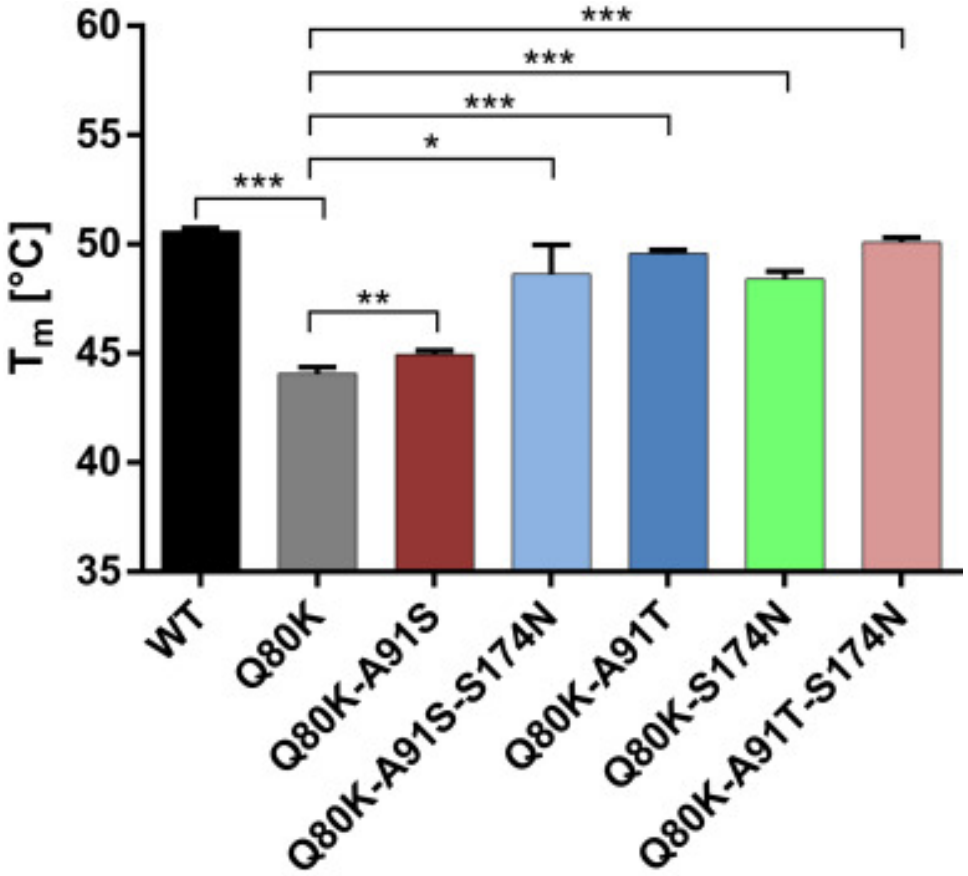

Figure 3.24: *Impact of NS3-Q80K and epistatic amino acid substitutions on the protease protein fold. T_m from purified NS3-4A protease WT and mutants are determined by fitting the sigmoidal melt curve, as shown in Figure 3.23, to the Boltzmann equation. Error bars represent the mean ± SD from at least three independent experiments: *$p \leqslant 0.05$, **$p \leqslant 0.01$, and ***$p \leqslant 0.001$, by two-sided t-test.*

Further, they hypothesized that the interactions among amino acids in NS3-4A might offset the replicative fitness decline caused by NS3-Q80K, possibly explaining its high prevalence. An *in vitro* Fluorescence Resonance Energy Transfer (FRET) based assay was conducted to evaluate the enzymatic activity and determine whether the stabilization of the protein structure affects the catalytic activity. The assay measures the cleavage of the viral polyprotein substrate at the NS4A/4B cleavage site. Results (Table 3.5 and Figure 3.25) showed that the NS3-Q80K protease had a lower catalytic turnover and affinity than the WT. The NS3-Q80K protease with the epistatic amino acid substitutions also shows reduced turnover rates but increased substrate affinity. However, compensatory effects were limited. Destabilizing the NS3-Q80K protease might enhance interaction with substrates and may not limit viral replication.

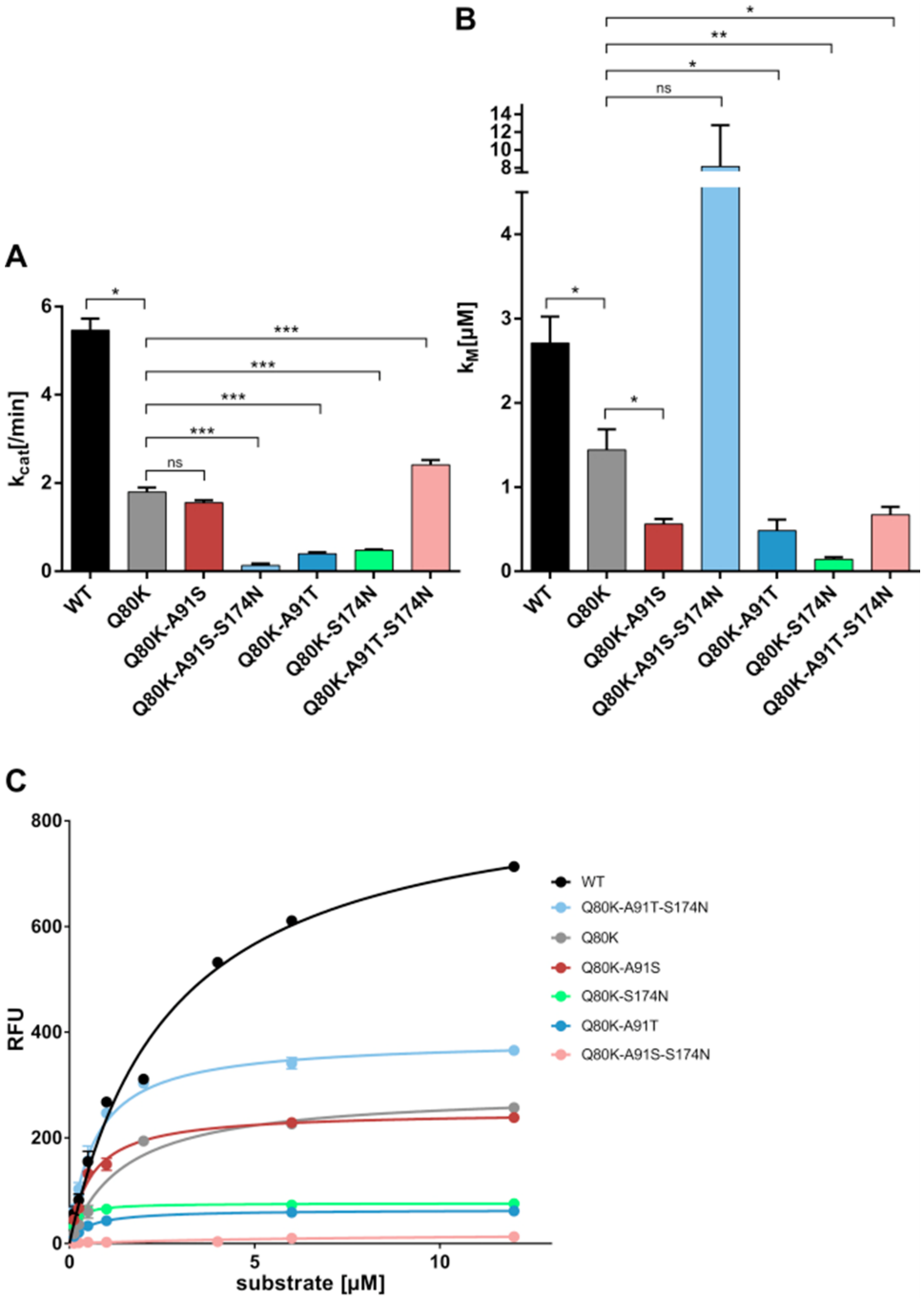

Figure 3.25: *Impact of NS3-Q80K and epistatic amino acid substitutions on the protease enzymatic function and reaction velocity. Reaction velocity and Michaelis-Menten kinetics as assessed from the purified protein of WT protease and NS3-Q80K mutants and the natural polyprotein substrate NS4A/4B. (A) reaction constants k_{cat} and (B) KM were calculated after fitting the nonlinear regression curve. Error bars represent the mean ± SD from at least three independent experiments: *$p \leqslant 0.05$, **$p \leqslant 0.01$, and ***$p \leqslant 0.001$, by two-sided t-test. C, kinetic progress curves with reaction velocities (RFU, relative fluorescence unit) for the NS3-4A protease and the natural polyprotein substrate NS4A/4B as assessed from the purified protein of WT protease and NS3-Q80K mutants.*

Results from the biochemical assays and MD simulations raised the question of whether the epistatic interaction might also contribute to viral replication by stabilizing the NS3-Q80K protease protein fold. To test this, the experimental partners introduced amino acid changes into the genotype 1a H77S.3 genome and examined their effect on viral RNA replication in transfected cells. The NS3-Q80K protease with epistatic substitutions showed minimal replication loss compared to the WT at 72 hours post-transfection, reaching WT levels by 96 hours (Figure 3.26). However, none of the epistatic amino acid substitutions increased replication compared to the parental H77S.3 genome or NS3-Q80K unexpectedly, indicating that these substitutions did not enhance viral replication.

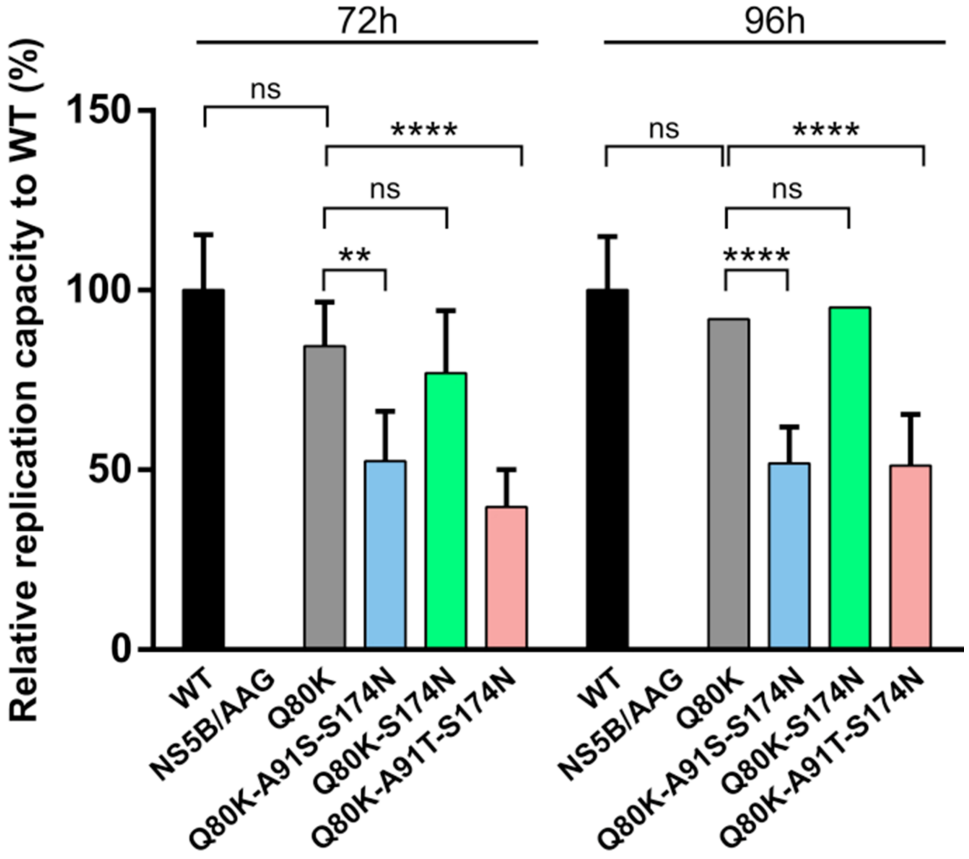

Figure 3.26: *Impact of epistatic amino acid substitutions on RNA replication of NS3-Q80K variants. The medium was collected and replaced at 8, 24, 48, 72, and 96 h after transfection of H77S.3/Gluc2A RNAs carrying the indicated mutations. Gluc activity was determined at 72 h (left panel) and 96 h in time (right panel). Results normalized to the 8h GLuc activity represent the mean of triplicate samples and multiple experiments. Error bars represent the mean ± SD from at least three independent experiments: *$p \leqslant 0.05$, **$p \leqslant 0.01$, ***$p \leqslant 0.001$, and ****$p \leqslant 0.0001$, by two-way ANOVA. GLuc, Gaussia luciferase.*

3.3.3 DISCUSSION

Compensatory mutations play a crucial role in the evolution of viral quasispecies, but structural limitations constrain their potential [265, 266]. Several studies demonstrated that amino acid changes in NS3-4A interact with NS3-Q80K, enhancing the stability of this polymorphism within viral variants [254, 258, 267–270]. The mechanisms driving these epistatic interactions in NS3-Q80K differ from those seen in compensatory mutations linked to resistance to protease inhibitors [271–274]. Despite their clinical significance in DAA treatment failures, the precise mechanisms behind these interactions have yet to be elucidated.

Our study showed epistatic amino acid substitutions stabilized the NS3-Q80K protein fold almost to the WT protease level. MD simulations showed changes in the total number of contacts between protease residues in NS3-Q80K and epistatic amino acid substitutions that reflect destabilizing versus stabilizing effects on the protease protein structure. This agrees with previous findings from [259], reporting variations in hydrogen bond occupancies from epistatic interactions in NS3-Q80K. Our structural models' total number of contacts agreed well with our protein stability measurements. This reflects the compensatory effect on the protein stability from epistatic amino acid substitutions in NS3-Q80K.

Our analysis of MD data has revealed that the Q80K mutation destabilizes the protein by impacting the N-terminal subdomain of the protease. This divergence in stability between the two domains alters the denaturation curves, with the N-terminal domain starting denaturation at lower temperatures due to destabilization by Q80K. This can be observed from a small shoulder around 45°C, exposing the bilobal protease fold. A more pronounced change occurs when Q80K is combined with additional mutations at positions 91 and 174, resulting in early denaturation followed by extended stability to higher temperatures due to epistatic substitutions. However, the stability in NS3-Q80K due to epistatic secondary substitutions also restricts the replication fitness of these variants. Our data suggest that replicative fitness may not be the main limiting factor in selecting NS3-Q80K variants during virus evolution. Moreover, in the protease activity assay using purified protein, none of the epistatic secondary substitutions enhanced the functional level of NS3-Q80K

protease, with some almost completely abolishing peptide substrate turnover.

Our findings suggest that protein stability is crucial in the persistence of NS3-Q80K variants. The amino acid substitutions associated with NS3-Q80K stabilized the protein structure, balancing against protease function. These interactions likely contribute to the persistence of NS3-Q80K through mechanisms not directly linked to RNA replication. Stabilizing a protein region containing a key epitope for immune responses, epistatic interactions may protect NS3-Q80K variants from T cell detection.

4

METAPROFI: ADVANCED DATA STRUCTURES FOR RAPID IDENTIFICATION OF FUNCTIONALLY RELEVANT GENETIC VARIANTS

This chapter is partly based on my publication, "MetaProFi: an ultrafast chunked Bloom filter for storing and querying protein and nucleotide sequence data for accurate identification of functionally relevant genetic variants" [15], of which I was the first author. My contributions to this work involved designing the project, conceptualizing and implementing the method, developing the software, analyzing the data, and writing the manuscript. This novel tool advances the field of sequence analysis by allowing for the efficient indexing and querying of both protein and nucleotide sequences. Unlike existing tools that are limited to nucleotide sequences, MetaProFi supports amino acid sequence indexing and allows nucleotide-based queries on these indices through an internal six-frame translation of the query sequences. This is particularly significant given that protein sequences are more conserved evolutionarily and often represent the most functionally relevant level of genetic information.

MetaProFi employs a Bloom filter-based approach [275], enhanced by several engineering optimizations such as shared memory systems, chunked data storage, and advanced compression techniques. These features enable MetaProFi to handle vast datasets with state-of-the-art performance. It maintains an excellent balance between memory consumption-to-speed ratio, making it a highly efficient tool for sequence analysis.

This chapter presents the background, related work, developed underlying methodologies, and the tool's application to large-scale nucleotide and protein sequence datasets. This section will explore how MetaProFi's capabilities can be leveraged to rapidly and accurately identify functionally relevant genetic variants.

4.1 BACKGROUND

As we explore indexing NGS data deeper, it becomes evident that the exponential growth in genomic data necessitates innovative solutions for efficiently storing and querying the extensive collections. The demand for tools capable of effectively managing and analyzing these datasets is more paramount than ever. While tools like BLAST [8] have been vital in enabling sequence searches by aligning query sequences against databases, they face significant challenges in scaling to the size of modern genomic databases, such as ENA. These traditional methods also require assembled genomes and cannot be applied to sequencing reads, as the results would contain only the matches within a single read. As the limitations of traditional tools become more apparent in the context of modern genomic data, relying solely on sequence alignment is no longer sufficient. The challenge lies in scaling and the fundamental way these tools store and query data. Addressing these issues requires a shift toward more sophisticated data structures that can handle sequencing reads' vast diversity and volume without compromising efficiency. Such tools rely on two fundamental core data structures: exact and approximate membership query data structures. In the subsequent sections, we unravel their complexities and assess their effectiveness in managing the ever-expanding landscape of NGS data. We explore how various tools address these challenges and highlight how our method can efficiently process and analyze the enormous amounts of NGS datasets.

4.1.1 EXACT MEMBERSHIP QUERY DATA STRUCTURES

Exact membership query data structures help to determine with certainty (zero false positives) whether a particular element is present or absent in a specific set. Examples of exact membership query data structures include hash tables, binary search trees, tries, and linked lists. Hash tables [276, 277] utilize hash functions to map elements to specific buckets where the element is stored. When performing a membership query, the hash function calculates the bucket where the element should be located, allowing for efficient retrieval and comparison. The strength of hash tables lies in their average-case time complexity for lookups, which is $O(1)$. However, they require a good hash function to minimize colli-

sions—cases where multiple elements map to the same bucket because collisions can degrade performance and increase retrieval time.

Binary search trees (BSTs) [276, 277] organize elements in a hierarchical, sorted structure, where each node has at most two children. The left child contains elements smaller than the parent node, and the right contains larger elements. Membership queries in a BST are executed by traversing the tree, starting from the root, and comparing the queried element with the current node's value. This allows rapid member checks by efficiently narrowing down the search space with an average-case time complexity of $O(\log n)$. However, the performance can degrade to $O(n)$ if the tree becomes unbalanced, as in the case of inserting already sorted data.

Tries, also known as prefix trees [276–278], are specialized data structures used for dynamic sets of strings. Each node in a trie represents a single string character, and the paths from the root to a terminal node (leaf) represent complete strings. Membership queries in tries are highly efficient for strings because they are based on the string's prefix, with time complexity proportional to the length of the string ($O(m)$, where m is the string length). Tries suffer from significant memory overhead due to the many nodes involved.

Linked lists [276, 277] are linear data structures where each element (node) points to the next element in the sequence. Membership checks in linked lists are performed by traversing the list from the head node to the end, comparing each node's value with the queried element. This traversal results in an $O(n)$ time complexity, making linked lists less efficient for membership queries than the other structures mentioned. However, linked lists offer advantages in scenarios where frequent insertions and deletions are required, as these operations can be performed in constant time, unlike in others where elements need to be shifted. Each of these data structures offers a different balance of efficiency, memory consumption, and operational complexity, making them suitable for various applications.

Nevertheless, these data structures cannot be employed to represent extensive collections of NGS data: even though they are famous for their fast queries, they are computationally expensive and not directly space efficient. These data structures are often more suitable for small datasets, for example, in genome assembly processes.

4.1.2 APPROXIMATE MEMBERSHIP QUERY DATA STRUCTURES

The shortcomings of the traditional data structures become increasingly apparent when dealing with extensive collections of NGS datasets. Approximate membership query (AMQ) data structures emerged as a promising alternative to address these challenges [279]. To reduce the required time and memory, a probabilistic data structure can be used to summarize and deliver fast approximate answers. Probabilistic membership data structures aid in determining whether an item/element is present. Several such data structures are already available, e.g., Bloom filter [275], Cuckoo filter [280], Quotient filter [281], etc. These data structures differ in their ability to offer approximation or definite answers. They are used in streaming applications and database lookups before performing expensive operations [282–285]. Importantly, these data structures guarantee zero false negatives; thus, if such a data structure returns an answer of a non-existent key, the database fetch/read is not required, saving time.

AMQ data structures offer several advantages over their exact counterparts, particularly in managing extensive data collection. While traditional exact structures like hash tables and BSTs struggle to scale efficiently with the exponential growth of data, approximate structures excel in scalability. They optimize memory utilization without compromising query performance, making them well-suited for handling massive datasets. They are inherently robust to errors introduced during sequencing and processing, ensuring reliable query results. By employing probabilistic algorithms and space-efficient representations, AMQ data structures balance scalability, memory efficiency, query speed, flexibility, and robustness to errors. These make them the most widely used in the field of bioinformatics to manage the ever-growing data, as they have the potential to retrieve not only closely but also partially closely related sequences via approximation. Hence, we focus on AMQ structures in this work. Next, we will describe some of these data structures and the specific features that allow them to achieve this improved performance.

4.1.3 WIDELY USED AMQ DATA STRUCTURES

The state-of-the-art AMQ data structures widely used for indexing NGS data include Bloom, cuckoo, and quotient filters, which allow for storing and managing vast and intricate genetic sequencing information. Several reasons to choose these data structures include the ability to streamline storage, retrieval, and analysis processes in bioinformatics tools, facilitate deciphering biological complexities, and advance precision medicine. This section illuminates the fundamental role of these structures.

4.1.3.1 BLOOM FILTER

Bloom filter [275] (BF) (Figure 4.1) is a probabilistic set-membership data structure that stores the presence or absence of items/elements in a bit vector. BFs guarantee zero false negatives and allow insert and query operations. The data structure comprises a bit vector filled with binary values where zero indicates absence, and one indicates presence. We start by filling the bit vector with zeros; for each given string, we apply h hash functions and use the return value of hashing as an index in the bit vector to flip the zero in the corresponding index position to one.

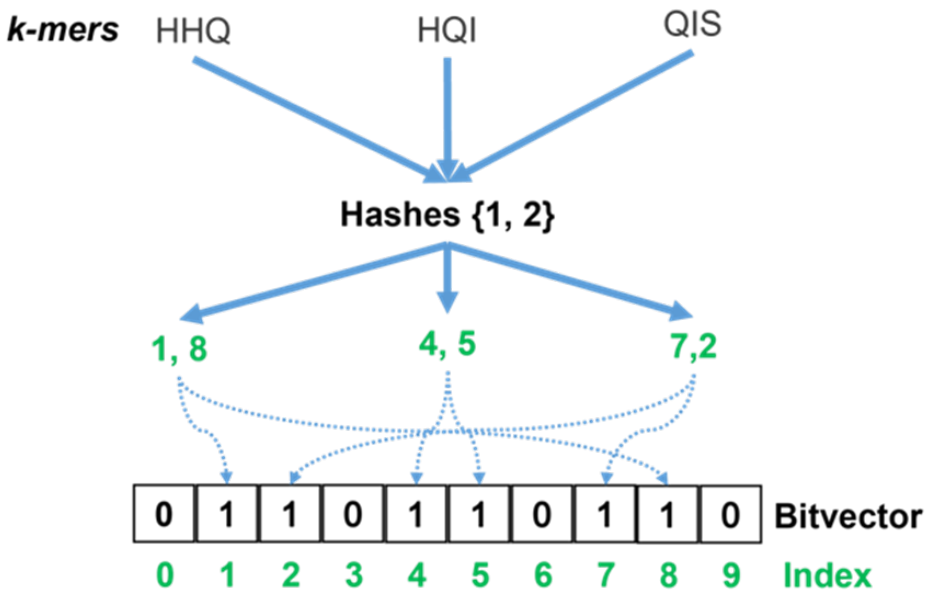

Figure 4.1: *Bloom filter data structure: Two hash functions are applied to each of the three strings, and the return values of hashing are used as an index in the bit vector to flip the zero in the corresponding index position to one.*

Both the insert and query operations have an $O(h)$ time complexity because h hash values need to be computed for every item regardless of the number of items already in the filter, unlike other approximate membership query data structures. Collisions (returning the same hash value for different strings) can be avoided by using large BFs and perfect hash functions, but in a realistic setting, this is not possible; thus, false positives arise. Their number can be reduced by increasing the size of the BF and the number of hash functions used. Assuming bit independence, i.e., the state of one bit in the BF does not influence or depend on the state of any other bit, the false positive rate can be calculated by,

$$p = \left(1 - \left(1 - \frac{1}{m}\right)^{hn}\right)^{h} \approx \left(1 - e^{-\frac{hn}{m}}\right)^{h} \tag{4.1.1}$$

where p is the false positive rate, h is the number of hash functions used, m is the size of the Bloom filter, n is the number of items inserted (or to be inserted) into the Bloom filter, and

$$\left(1 - \frac{1}{m}\right)^{hn} \approx e^{-\frac{hn}{m}} \tag{4.1.2}$$

is the probability that a specific bit is still 0 after inserting n items. An optimal number of hash functions can be calculated by,

$$h = \frac{m}{n} \ln(2) \tag{4.1.3}$$

and the size of the Bloom filter can be estimated by,

$$m = -\frac{n \ln(p)}{(\ln(2))^2} \tag{4.1.4}$$

Tools utilizing Bloom filters and their variants are discussed in Chapter 4.2.

4.1.3.2 CUCKOO FILTER

The Cuckoo filter (CF) [280] is an approximate membership query data structure that supports dynamic inserts, deletes, and lookup operations using partial-key Cuckoo hashing, a variant of Cuckoo hashing [286] that stores only fingerprints - a bit string derived from the item using a hash function instead of storing the keys.

Cuckoo hashing employs an open-addressing technique and uses a hash table consisting of an array of buckets where each item has two candidate buckets determined by two hash functions. The insert operation takes the latest item and hashes it with the two hash functions, then takes the resulting hash values, looks up the respective locations in the hash table or the array, and checks if either of the two buckets is empty; the algorithm then inserts the item to that free bucket. If neither of the buckets is free, then a bucket is chosen from the two, and it kicks out the existing item, inserts the latest item, and then relocates the kicked-out item to its alternate location. This process may repeat until a vacant bucket is found or until a maximum number of relocations is reached. If no vacant bucket is available, this hash table is too full to insert new items. The lookup operation checks both buckets to see if either contains the item in the query; if so, it returns the item and is successful. Similarly, the delete operation checks the target item's potential positions in the hash table. If the item is identified in either position, it is removed from the hash tables. If the item is not found in either of the hash tables, additional measures such as rehashing or probing neighboring areas may be applied. The same can also be used during the lookups. Cuckoo hashing has high space occupancy as it keeps relocating earlier items placed in the bucket to make room for the new items. Insert and delete operations take $O(1)$. However, in worst cases, it might require rehashing, leading to $O(n)$, where n is the number of items. Similarly, the lookup operations take $O(1)$ in the worst case.

Cuckoo filters use a multi-way associative cuckoo hash table to offer fast lookups and high table occupancy. Partial-key cuckoo hashing hashes and stores a constant-sized fingerprint ($f(x)$), unlike cuckoo hashing, which stores the items. In the cuckoo filter, the basic unit of the hash table is called an entry, and each entry can store only one fingerprint. The hash table consists of an array of buckets where a bucket can

have multiple entries. The insert operation, however, cannot rely on the cuckoo hashing technique as there is no way to restore and rehash the items to find their alternate locations, because it stores only the fingerprints. However, partial-key cuckoo hashing can derive the index of item x and its alternate location through the following two hash functions [286, 287].

$$h_1(x) := \text{hash}(x) \tag{4.1.5}$$

$$h_2(x) := h_1(x) \oplus \text{hash}(f(x)) \tag{4.1.6}$$

By defining these two hash functions, the insert operation can directly calculate the alternate location from the current location and the fingerprint stored in the bucket. Therefore, this operation only uses the information in the table without retrieving the original item. Using partial-key cuckoo hashing, new items can be inserted dynamically. The lookup and delete operations follow the same principle as the cuckoo hashing; however, they might have an overhead because they depend on the size of the buckets.

Cuckoo filters have lower space overhead than space-optimized Bloom filters for applications that require low false positive rates. Previous attempts to extend Bloom filters to support deletion have degraded space or performance. The Cuckoo filter addresses these limitations and outperforms other data structures that extend Bloom filters to support deletions in both time and space [280]. In Cuckoo filters, the size of the fingerprints is inversely proportional to the acceptable false positive rate (FPR). The lower the desired FPR, the longer the fingerprint size needs to be in order to reject false queries. The fingerprint size also grows logarithmically with the number of items in the filter. As a result, the space required for per-item overhead is higher for larger hash tables. Tools like NGSReadsTreatment [288] use the Cuckoo filter to identify and remove redundant reads in NGS datasets.

4.1.3.3 QUOTIENT FILTER

A quotient filter (QF) [281] is another space-efficient probabilistic data structure. QF stores only a part of the item's hash fingerprint along with additional metadata bits. QF supports insert, lookup, and delete

operations. QF data structure is a bit-array of length m with additional metadata bits. The hash function generates a p-bit fingerprint. The p-bits fingerprint is split into remainder *(r)*, which consists of the least significant bits of the p-bits fingerprint, while the quotient $(q = p - r)$ comprises the most significant bits. The size of the QF is $m = 2^q$ slots. Upon initialization, all the bits in the QF are set to zero. The hash values are split into quotient and remainder. The quotient is used to identify the slot position while the remainder is stored in the slot in the QF. For example, if the size of the QF is eight and the output of a hash function is 8 bits *(p)* long, then solving $m = 2^q$, the quotient is 3, and the remainder is 5.

Before looking into the operations allowed by the QF, let's define some QF terminologies. QF defines a *canonical slot* as the original slot to which the quotient of the hash value points. A set of fingerprints with the same quotient is called a *run*. Each slot is associated with three metadata bits: *is_occupied*, *is_continuation*, and *is_shifted*. The metadata bit *is_occupied* is set when a slot is canonical, and *is_continuation* is set when a slot is occupied but not by the first remainder in a *run*. Finally, *is_shifted* is set when the remainder in a slot is not in its *canonical slot*. A *cluster* is a contiguous sequence of *runs*; it contains a canonical entry at the beginning and does not contain an empty slot between the *runs*. A *super cluster* is a contiguous sequence of clusters with no empty slots.

An item is hashed to perform an insert operation on an empty QF, and the quotient and remainders are calculated. The quotient is then used to identify the slot position; in other words, the quotient serves as the index in the QF. The remainder is stored in the corresponding slot, and the *is_occupied* metadata bit is set to 1. During the insert operation, if multiple items' hash values result in the same quotient, it is denoted as a soft collision, and the appropriate metadata bits are set to 1. For example, when a slot is already occupied and the new items' quotient points to the same slot, QF uses linear probing to find the next empty slot and adds a reminder to it. Then, the *is_continuation* and *is_shifted* metadata bits are set to one, indicating that the slot is occupied but not by the first remainder in a *run* and that the remainder in the slot is not in its *canonical slot*.

The lookup operation involves hashing the item, splitting the hash into quotient and remainder, then looking for the remainder in the respective

slot and verifying that the *is_occupied* metadata bit alone is set to one. If the metadata bit is unset, the query item is not present in the QF. The item may be present in the QF if it is set to one. The uncertainty is due to the false positives caused by hard collisions (same hash values for multiple items), the number of items already inserted in the QF, and the size of the fingerprints. However, suppose the *is_occupied* metadata bit is unset, and the other bits are set. In that case, we perform the same linear probing operation we applied during the insert operation to find if the query item is present in the QF.

The delete operation follows the same principles as the insert operation, but after removing the item from the QF, the *super cluster* entries are shifted backwards.

All operations have an $O(1)$ time complexity in the average case. In some operations, the time increases with the size of the QF. Since additional metadata is required for every entry, 25% more space is needed. Also, when the occupancy of the hash table exceeds 75%, there is a significant drop in performance because the likelihood of soft collisions increases, which leads to more frequent linear probing when performing insertions and lookups. On the other hand, QF is sensitive to false positives due to collisions. The false positive rate depends on the size of the fingerprints and the number of items inserted.

4.1.3.4 COUNTING QUOTIENT FILTER

Counting quotient filter (CQF) [289] represents a significant advancement over traditional QF by addressing several limitations. CQF achieves this by incorporating counters associated with each filter entry, effectively addressing collision management issues, and enhancing accuracy by maintaining item counts within the filter. This integration reduces the likelihood of false positives, thereby supporting the reliability of AMQ. CQFs restructure the metadata bits to speed up the operations (insert and lookup) and maintain reliable performance up to a load factor of 95% of the filter. This restructuring optimizes lookup efficiency and saves space by reducing the number of metadata bits allocated per item. Additionally, CQF implementations leverage rank and select [290] optimizations within the metadata structure to further increase efficiency compared to traditional QFs. By utilizing these optimizations, CQFs

streamline metadata operations, requiring only 2.125 bits for metadata storage per item compared to the three metadata bits consumed by QFs.

4.1.3.5 SELECTING THE OPTIMAL FILTER

Choosing between Bloom, cuckoo, and quotient filters for NGS data indexing depends on the application's specific requirements. Though the above-discussed data structures offer similar operations, the cuckoo and quotient filter allow deleting keys, which can be helpful when indexing extensive collections of NGS data, especially during sample replacement, as this offers the native functionality to remove samples instead of rebuilding the entire index just because of a single sample change. Despite these filters presenting lower bounds compared to Bloom filter (BF), BFs remain a valuable data structure to use for indexing extensive collections of NGS data because of the following,

1. BFs provide a space-efficient way to store items, allow for fast membership queries, and are well-suited for large datasets.

2. The lookup is straightforward in BF as the keys are not moved around, unlike in CF and QF, where keys are moved around due to collisions. This is particularly valuable when indexing extensive collections of samples, as BFs maintain consistent insertion patterns. In contrast, CF and QF require probing techniques that can alter the insertion order due to collisions, potentially complicating large-scale indexing and querying.

3. False positives do not affect the BFs in the case of sequence indexing, as the query sequences are usually large. The false positive rate for the whole query sequence is low compared to that of an individual k-mer, making BFs more reliable for large-scale queries.

4. Insertion and lookup times in BF are constant regardless of the number of items already present in the filter. In contrast, CF and QF involve key movement upon collisions, which can lead to variable performance as the filter fills up.

5. BFs offer a simpler query process compared to CFs or QFs, particularly when indexing extensive collections of NGS data. Their bit-array representation enables the use of various compression

techniques, allowing for further optimizations that enhance the efficiency and scalability of the indexing process while optimizing storage utilization.

In addition to the above points, and as previously described, the tools developed as part of the thesis aimed not only to index extensive collections of NGS data but, more importantly, to address a significant gap in the field of protein sequence indexing. While several tools can index NGS data, none supports amino acid sequence indexing, which creates a notable limitation. We aim to bridge this gap by proposing the first tool capable of indexing both amino acid and nucleotide sequence data. This tool also allows for querying amino acid-based indexes with nucleotide queries by internally performing a six-frame translation, thus filling a critical gap in how protein sequences can be stored and queried.

4.2 RELATED WORK

Before introducing the tools and methods that were implemented, here is an overview and summary of the state-of-the-art tools, including their implementations and the challenges they aim to address in nucleotide sequence indexing.

4.2.1 SEQUENCE BLOOM TREE

The existing full-text indexing data structures, such as Burrows-Wheeler transform [76], FM-index [79], or others, are currently unable to mine data from databases like ENA and SRA containing several petabases. None of the existing approaches can match a query sequence q that spans many short reads. Sequence Bloom Trees (SBT) is a tool developed to address these issues [291]. The tool is used to identify all samples in a dataset containing a given query sequence q with reduced memory requirements. SBT index is not limited to searching for known sequences and can be built efficiently and stored in a limited additional space. In addition to insertion and query operations, SBTs allow deletions and the addition of new experiments.

An SBT (Figure 4.2) is a rooted binary tree built by repeatedly inserting sequencing experiments. The first step involves the construction of

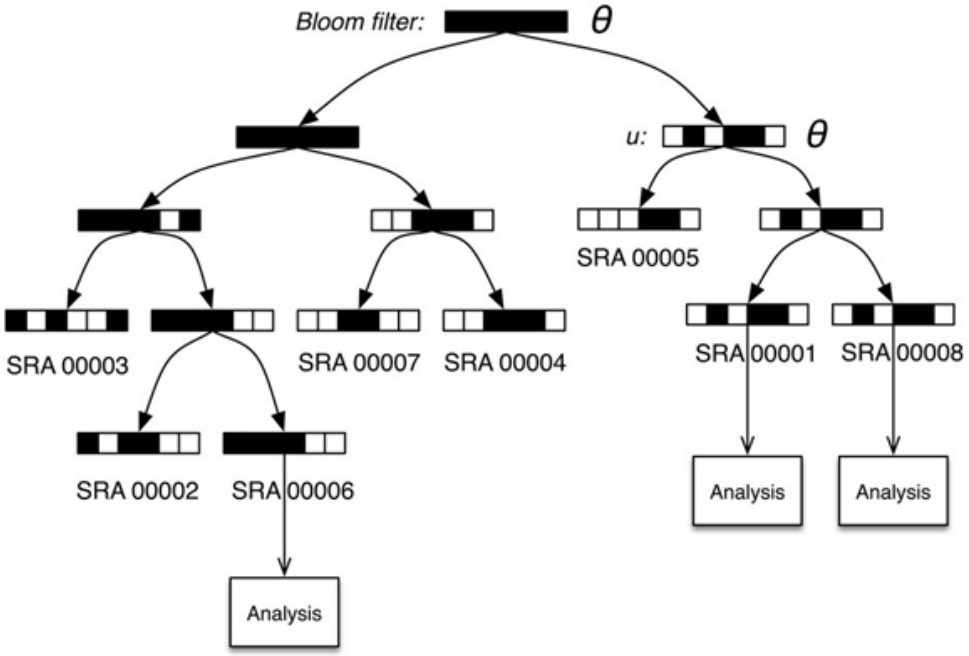

Figure 4.2: *(Source: (Solomon et al., 2016 [291])) Overview of sequence Bloom tree. Each node in the structure includes a bloom filter that stores the k-mers identified in the associated sequencing experiments. The parameter θ represents the proportion of k-mers that must be present at each node to search its subtree. The SBT returns the experiments most likely to include the query sequence, allowing for further analysis.*

Bloom filters (BF) for *k*-mers present in the sequencing experiment. The algorithm then navigates the tree, inserting the sample based on the similarity of its Bloom filter to the existing nodes. To insert a new experiment s into the tree T, a Bloom filter ($b(s)$) is computed for the *k*-mers in s. The tree T is then traversed from the root towards the leaves, and s is inserted at the bottom of T. If a node u has one child, insert a new node representing s with $b(s)$ as the second child of u. If u has two children, compare $b(s)$ with the Bloom filters of the left and right children and choose the child with the more similar filter based on Hamming distance, and this becomes the current node, and the process is repeated. If u has no children, it represents another sequencing experiment s'. In that case, a new union node v as a child of u's parent is created. Now, v has two children, u, and a new node representing s [291].

SBTs' insertion process follows a greedy approach, aiming to group similar experiments by their Bloom filters to reduce filter saturation and query time. Filter saturation is a big challenge when scaling SBTs to extensive collections of sequencing experiments or experiments containing diverse samples because BFs at higher levels of the tree tend to have

more bits set to 1 in such cases, thus increasing the false-positive rates and, in turn, increases the query time.

Given a query sequence q, the k-mers are extracted, and the tree is traversed from the root. A threshold is defined between 0 and 1, which governs the query's error tolerance. Higher thresholds expect a greater number of query k-mers to be present. During tree traversal from the root at each node u, the Bloom filter in node $b(u)$ is queried for the presence or absence of each k-mer. If more k-mers are found than the defined threshold, the traversal continues to all children of u. If fewer k-mers are found, then the subtree is pruned, thus reducing the query time directly at the higher levels of the tree. Only the tree topology and the current filter are stored in the memory, leading to reduced memory requirements.

The authors benchmarked SBT on a collection of 2652 human RNA-seq samples containing blood, brain, and breast samples from SRA. An SBT index was constructed and required 200 GB (2.3% of the size of the original sequencing data) of storage with compression. It took approximately three days to build the BFs using 20 threads, and the construction of SBT with compression took roughly 34 hours using one thread. A single transcript query from the dataset required 239 MB of RAM and took approximately 20 minutes on a single core. The authors assessed batches of 100 queries and found that SBT was an estimated 4056 times faster than SRA-BLAST [292] and STAR [293] alignment-based methods.

4.2.2 SPLIT SEQUENCE BLOOM TREES

Traditional approaches like the SBT offer solutions but struggle with scalability when dealing with petabases of data. The split sequence Bloom tree (SSBT) [294] method develops as an enhancement to the SBT and is designed to address the challenges of storage efficiency and query performance in large-scale sequencing datasets.

SSBT (Figure 4.3) builds upon the SBT by introducing a new way of managing Bloom filters to reduce redundancy and improve query efficiency. In an SSBT, each node in the binary tree contains two distinct Bloom filters: a similarity filter and a remainder filter. The similarity filter captures k-mers that are universally present across the experiments

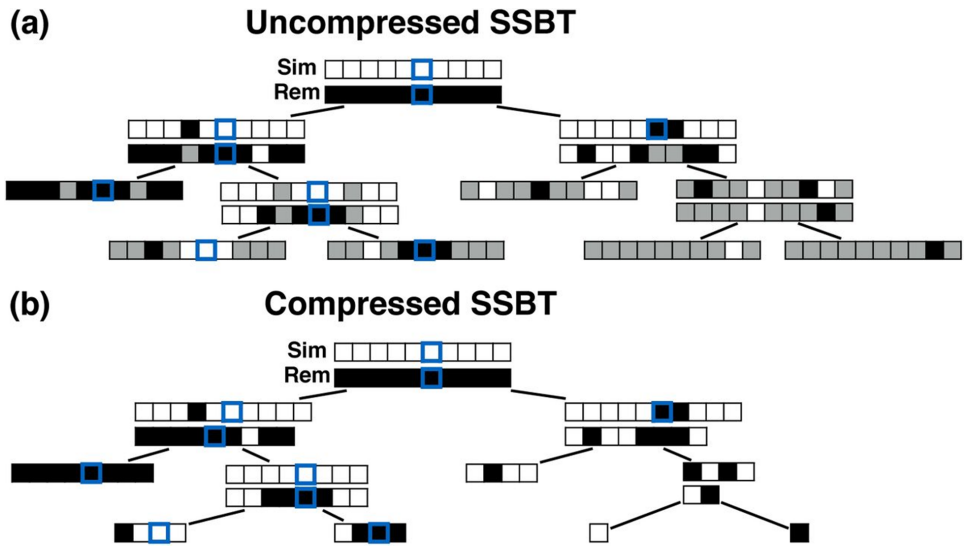

Figure 4.3: *(Source: (Solomon et al., 2018 [294])) An overview of uncompressed and compressed SSBTs, where black represents a bit value of one, white represents a bit value of zero, and grey bits indicate non-informative bits whose values are determined by the parent filter.*

within the node's subtree. In contrast, the remainder filter stores the *k*-mers that appear in at least one experiment but are not universal. This dual-filter system reduces filter saturation. As the tree is traversed, the SSBT's structure allows for efficient pruning of subtrees. The method checks the Bloom filters at each node when querying the tree with a sequence. If the query's *k*-mers are present in the similarity filter, the traversal continues down the tree. If not, the subtree is pruned, skipping checks and reducing the query time.

The performance of SSBT was benchmarked using a collection of 2652 human RNA-seq samples (the same as SBT). When comparing the performance of SSBT to SBT, SSBT demonstrates significant improvements in storage efficiency and query performance, albeit at the cost of increased build time and memory usage. SSBT index construction took 78 hours, considerably longer than the 18 hours required for SBT. The compression process was also slightly more time-consuming for SSBT, taking 19 hours compared to SBT's 17 hours. However, these are offset by SSBT's reduction in storage requirements. The final compressed size of the SSBT index was 39.7 GB, significantly smaller than the 200 GB required for the SBT index. In terms of query performance, SSBT outperforms SBT substantially. For queries involving transcripts per million (TPM) thresholds of 100, 500, and 1000, SSBT consistently computed

the results in approximately 3.6 to 3.8 minutes, compared to the 19.7 to 20.7 minutes needed by SBT, demonstrating a fivefold improvement in query speed. Although SSBT's construction process is more complex and requires more memory and time than SBT, its query performance and storage efficiency make it a more effective solution for managing and querying large-scale sequencing datasets.

4.2.3 ALLSOME SEQUENCE BLOOM TREES

AllSome sequence Bloom trees (SBT) [295] build upon the SBT and introduce key construction and query efficiency improvements. In the original SBT, each node is represented using a single bit vector compressed with RRR [296], facilitating efficient look-ups without decompression. Inserting a new bit vector into the SBT involves locating the appropriate child subtree based on the Hamming distance or creating a new root if necessary. Queries are executed by recursively traversing the tree, loading nodes from the disk as needed, and matching queries at the leaf nodes.

The AllSome SBT (Figure 4.4) refines the SBT method by incorporating three key improvements. Firstly, it utilizes a non-greedy construction method through agglomerative hierarchical clustering, improving k-mer matches' localization. Instead of the original greedy insertion strategy, this method merges pairs of SBTs based on the smallest Hamming distance between their root bit vectors, leading to a more efficient tree organization. This improved structure facilitates faster query resolution by ensuring that related k-mers are clustered. Additionally, the AllSome SBT introduces a new node representation using two-bit vectors: $B_{all}(u)$ and $B_{some}(u)$. $B_{all}(u)$ captures bits present in all leaves of the subtree rooted at node u, excluding those found in the leaves of the parent node, while $B_{some}(u)$ contains bits present in some but not all leaves. This representation reduces the bitwise operations and disk accesses required during queries by resolving many k-mer matches earlier. For large-scale queries, where a query might consist of millions of k-mers, the AllSome SBT implements a dedicated large query algorithm. Instead of checking each k-mer individually, this algorithm performs bitwise comparisons using the entire query's bit vector (AllSome SBT constructs a Bloom filter for the entire query itself). This approach is significantly more efficient

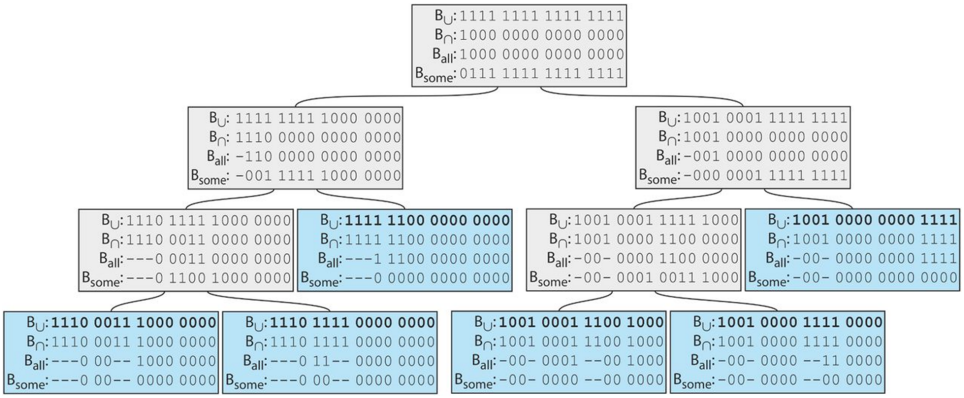

Figure 4.4: *(Source: (Sun et al., 2018 [295])) Example AllSome SBT on a set of sequences. Leaves are shown in blue and internal nodes in gray.*

for handling large queries than the traditional method, which becomes a bottleneck due to the time required for numerous individual checks.

The AllSome SBT demonstrates improvements over the SBT in several areas benchmarked using the same collection of 2652 human RNA-seq samples as in SBT. During construction, AllSome SBT reduces the time required by 52.7%, taking 26 hours and 3 minutes to build the internal nodes compared to 56 hours for SBT. The final disk space usage is also lower, at 177 GB for AllSome SBT compared to 200 GB for SBT. However, it requires more memory and disk space during the construction, with peak memory usage of 908 MB compared to 726 MB for SBT and temporary disk space of 2469 GB versus 1235 GB for SBT.

For regular queries, AllSome SBT achieves a 39%-85% reduction in runtime compared to SBT. For instance, it processed 198,074 queries in 463 minutes, whereas SBT required over two days (3082 minutes). The large query algorithm in AllSome SBT further accelerates performance, up to 155 times faster than SBT, and efficiently handles large queries. However, AllSome SBT uses more memory for large queries, with maximum usage of 63 GB compared to SBT's 22 GB for 198,074 queries.

The combination of clustering and the AllSome representation together enhances performance. Clustering alone improves node look-up efficiency by 36.5% and reduces query times by 19%-32%. Meanwhile, the AllSome representation is particularly effective for queries targeting many leaves, with reductions in node look-ups of up to 27.4% for high-hit queries.

4.2.4 HOWDE-SBT

HowDe-SBT is a further improved representation of SBT [297]. The authors present an alternative way to partition and organize the data in an SBT, thereby improving the compression ratio and faster querying. They also propose a culling procedure to remove non-informative nodes from an SBT, thus creating a non-binary forest. An empty HowDe-SBT (Figure 4.5) has a tree topology T with a bijection between its leaves and the Bloom filters or the bit vectors of the datasets that will be created and assigned to the internal nodes.

At each node u, two bitvectors, B_{det} and B_{how}, are constructed. B_{det} represents the positions that have the same value across each of the leaves of node u, while B_{how} is informative only for the positions in B_{det} and defines them as follows:

$$\mathrm{B}_{\mathrm{det}}(u) \triangleq \mathrm{B}_{\cap}(u) \cup \mathrm{B}_{\cup}(u) \tag{4.2.1}$$

$$\mathrm{B}_{\mathrm{how}}(u) \triangleq \mathrm{B}_{\cap}(u) \tag{4.2.2}$$

$B_{how}(u)$ reflects the bits that are common (the same value) in all BFs corresponding to the leaves of node u. $B_{det}(u)$ is the union of $B_{how}(u)$ and any additional bits that are relevant, providing a detailed view of the bit positions associated with node u. Computing B_{det} and B_{how} for each node allows for efficient querying. Each k-mer in a query q is hashed (only one hash function), and the respective positions in the BFs are determined, and these positions are called unresolved positions. Two zero-initialized counters are maintained, one for positions determined to be 1 (present) and one for the positions determined to be 0 (absent). The algorithm follows a recursive search starting at the root of T. When comparing a k-mer in q against a node u, each unresolved position that is 1 in $B_{det}(u)$ is removed from the unresolved list, and the corresponding bit in $B_{how}(u)$ determines whether to increment the present or absent counter. If the present counter reaches a set threshold value (between 0 and 1), the leaves of node u are added to the list of matches, and the further search of the node's subtree is pruned. If the absent counter exceeds the set threshold, then u's subtree search is pruned because the query cannot match any of node u's leaves. When neither condition

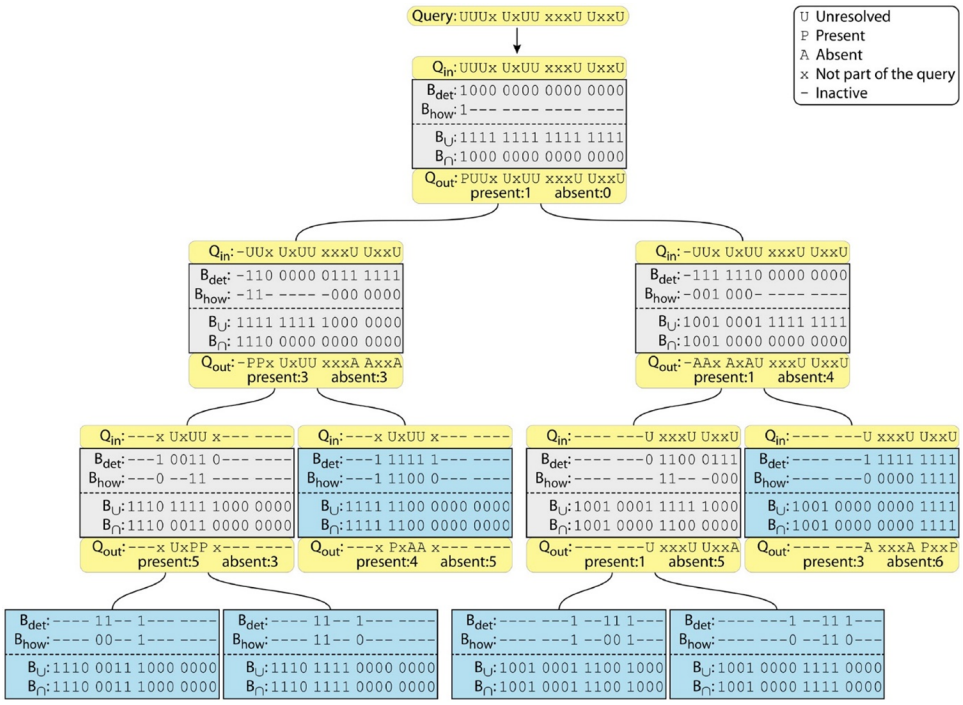

Figure 4.5: *(Source: (Harris et al., 2019 [297])) Example HowDe-SBT on a set of sequences. Each box represents a tree node, with leaves in blue and internal nodes in gray. The yellow boxes illustrate the processing of a nine k-mer query with a threshold* θ*. The initial query is at the top, with bits for query k-mers marked as U. Bits are marked as present (P) or absent (A) and are removed from further processing if counted in the totals.*

holds, the algorithm continues to search the u's children. The query unresolved list becomes empty when it reaches the leaf, because the bit vector B_{det} is all ones at a leaf, and therefore, the algorithm terminates.

The algorithm is optimized to achieve a better compression ratio and faster querying by identifying and removing inactive bit positions from B_{det} and B_{how} bit vectors that will never be queried during the search operation. Inactive positions are those already determined at a node or its ancestors and are not queried. The active bits form compressed bit vectors, further compressed using the RRR compression algorithm. These compressed bit vectors constitute the final index.

HowDe-SBT was benchmarked with SSBT and AllSsome SBT using the same human RNA seq dataset described in SBT, and they removed samples from the dataset that did not contain reads longer than k. HowDe-SBT took 9 hours to build an index, and this does not include the time taken to construct the Bloom filters for each read sample or a read file in the dataset. The authors mention that using multiple threads to build the Bloom filters took several days. A total of 616 GiB of storage was consumed in the build process, where the final index took 14 GiB, showing that the index was constructed in less than 36% of the time and with 39% less storage compared to other approaches. The query performance was compared by creating four types of queries: a single transcript, a batch of 10 transcripts, a batch of 100, and a batch of 1000. Comparing the query performance, HowDe-SBT shows over a 5x speedup on single-transcript batches with a peak memory usage of <1.3 GiB for all batches for all tools.

4.2.5 MANTIS

Before discussing Mantis [298], let's first look into color-aggregative indexing to help better understand the tool.

Color-aggregative methods-based indexing involves grouping or aggregating similar k-mers based on specific properties. The advantage of color-aggregative methods is that a k-mer that appears in many samples is represented only once in the union set, thus reducing the redundancy but introducing the need to store additional color data along with the k-mer data. This was first introduced by Iqbal *et al.* in the Cortex tool [299]; it was developed for *de novo* assembly and detection and genotyp-

ing of simple and complex genetic variants. This tool utilizes a simple associative data structure to assign colors to *k*-mers by extending traditional De Bruijn Graphs (DBG) concepts. A DBG is a directed graph that represents the overlaps between adjacent symbols. In sequencing and *k*-mer indexing, the *k*-mers form nodes in the graph, while the edges represent overlapping adjacent *k*-mers. This concept was extended through Cortex tool by associating each *k*-mer with a color, denoted as a colored DBG. The color can represent various attributes; however, in our context, it usually denotes the origin or source of the sample or the sequence. In a traditional DBG, the graph would only show the *k*-mer overlaps without additional contextual information about the species origin of each *k*-mer. In contrast, a colored DBG enhances this concept by associating each *k*-mer with a specific color that typically represents the sample or species origin. This coloring allows *k*-mers to be differentiated from different samples or species, and is particularly useful for identifying genetic variants like SNPs and other short variants. These variants can be visualized in the graph through pairs of short paths sharing common starting and ending nodes that indicate regions where *k*-mers from different sources converge or diverge, thus highlighting potential SNPs or other genetic variations [300].

Mantis is used to store and query large-scale sequencing datasets. It leverages the colored De Bruijn Graph (cDBG) representation to organize and index sequences. Mantis builds upon Squeakr [301], a tool that is a wrapper for constructing a CQF. Squeakr's role involves parsing the input files, extracting *k*-mers, and inserting them into a CQF. In Mantis (Figure 4.6), the color assigned to each *k*-mer within a cDBG corresponds to the experiments where that *k*-mer is present. An exact CQF maintains a table linking each *k*-mer to a unique color identifier (ID). In contrast, another table maps color IDs to the specific experiments containing the corresponding *k*-mer. To build an index, individual CQFs are constructed for each input sample using Squeakr, while the tool also utilizes an off-the-shelf compressor to store the bit-vectors. These individual CQFs are merged into a unified CQF structure, which stores color IDs instead of *k*-mer counts.

The authors assert that the existing approaches, such as the SSBT, encounter challenges related to the fundamental limitations leading to prolonged build and query durations, suboptimal space utilization,

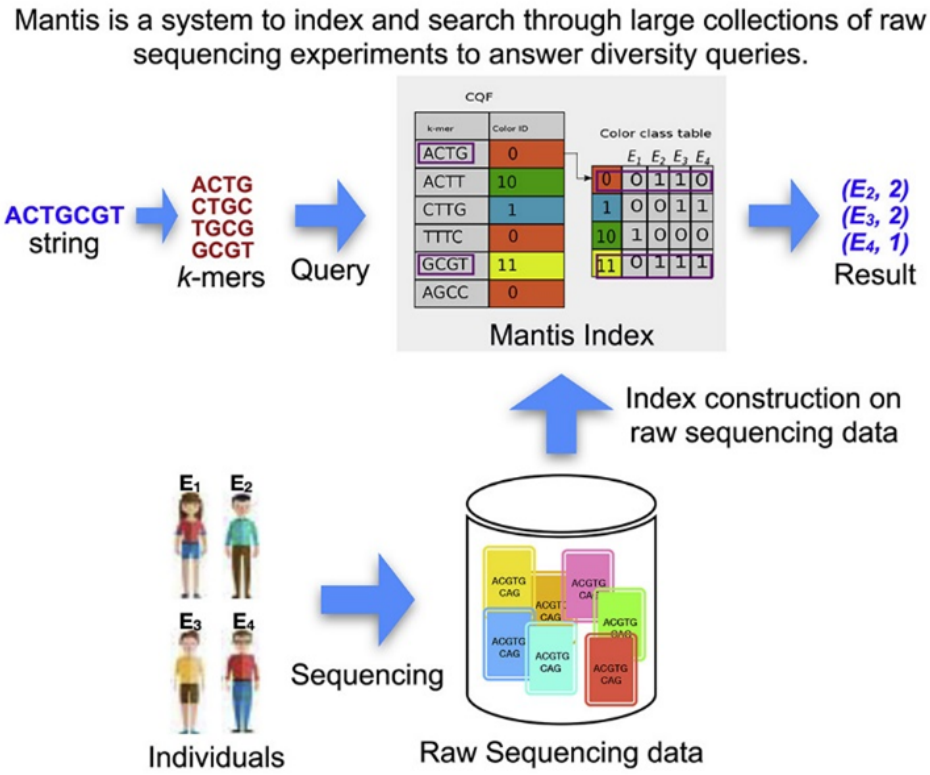

Figure 4.6: *(Source: (Pandey et al., 2018 [298])) An overview of Mantis indexing and query search.*

and the potential for many false positives. Through Mantis, the authors address these issues by employing an optimization to index thousands of raw-read experiments, facilitating large-scale sequence searches. In their evaluation, the authors find that constructing the index with Mantis is 6-108 times faster and yields a 20% smaller index than SSBT.

4.2.6 BITSLICED GENOMIC SIGNATURE INDEX

Unlike the color aggregative methods, which focus on reducing redundancy by grouping similar *k*-mers across multiple samples, the BItsliced Genomic Signature Index (BIGSI) [302] employs the *k*-mer aggregative approach where the dataset is processed sample by sample rather than pooling all *k*-mers from all samples to build an index. After processing, the individual *k*-mer indices from each sample are aggregated using various techniques to form a comprehensive index for the entire dataset.

SBTs were the first to offer a scalable approach to indexing extensive collections of datasets. However, they are limited by a scaling dependence on the union of the indexed dataset's total number of *k*-mers. This means that as the diversity of samples or *k*-mers increases, the efficiencies offered by these tools are diminished. In contrast, BIGSI addresses this limitation using a fixed-size binary signature for each dataset. BIGSI was the first method to use BFs to represent an extensive collection of 447,833 bacterial and viral whole-genome sequence datasets from ENA. The BIGSI index required 1.5 TB of storage—only 1% of the original 170 TB size—to represent all the *k*-mers present, offering a more scalable solution for managing large and diverse genomic datasets.

BIGSI approaches this as a document retrieval problem by internet search engines that treat *k*-mers as the search or query terms, and the documents are the read datasets or assembled genomes. In this approach (Figure 4.7), a BF matrix is constructed where each column represents a sample, represented by a filter of *k*-mers within a dataset, and the rows correspond to the index of the *k*-mers. From the query point of view, the BF matrix allows efficient ($O(1)$) retrieval as the index of every *k*-mer from every sample appears as a consecutive block. Therefore, they can be easily retrieved as a single slice (bitslice) of a bit vector. BIGSI performs a bitwise AND operation between the retrieved bitslices to obtain a bit-vector representing the presence or absence of the query

in each sample in the indexed dataset. Since BIGSI builds BF for every sample and then aggregates them in a matrix, adding new samples is as simple as creating a BF for the latest sample and then appending the new BF to the end of the matrix as a new column. However, such appending requires modifying every key in the BIGSI. They construct an index for new samples in batches and perform a merge operation instead of building an entirely new index.

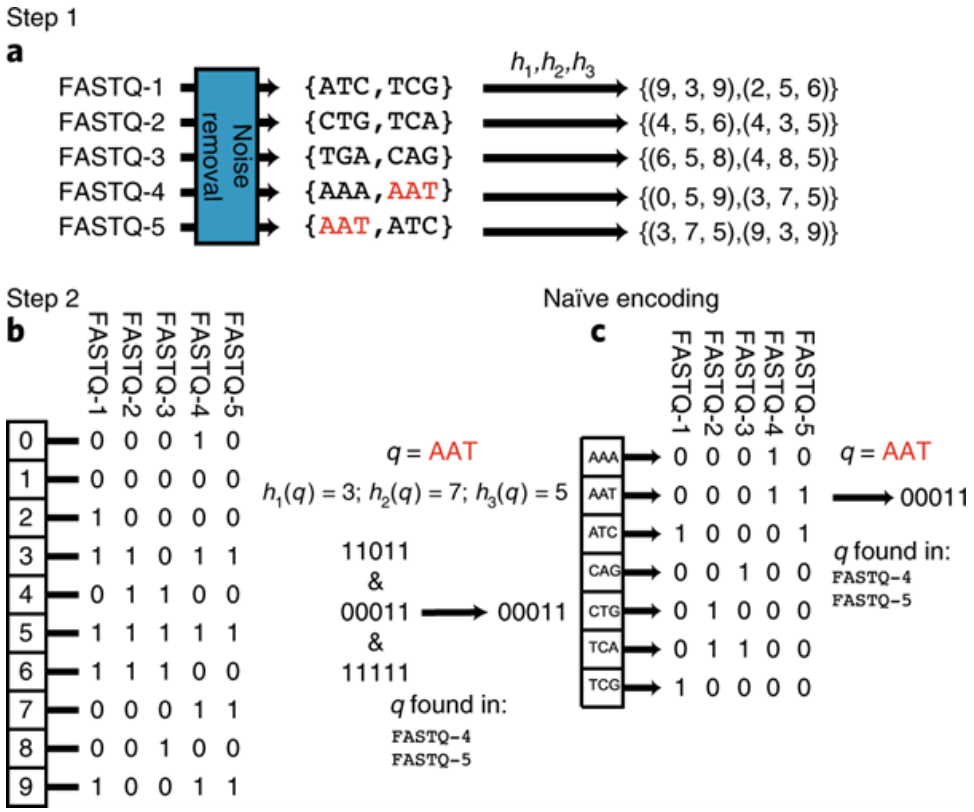

Figure 4.7: *(Source: (Bradley et al., 2019 [302])) An overview of construction and query of a BIGSI.*

The false positive rate of a BF is controlled by the choice of the size of the BF and the number of hash functions used. In BIGSI, the authors create indexes with relatively high per-k-mer error rates to achieve a better compression rate, but this requires a longer minimum query sequence length. The authors mention that the false positive rate per Bloom filter can be as high as 0.3, and by requiring a longer minimum query sequence (for example, 61), the false positive rate per query k-mer decreases exponentially with each additional unique k-mer in the query. Assuming independence of k-mer and Bloom filters, the expected number of false discoveries (V) for a query (q) of length L can be expressed as

$q = E[V] = Np^L$, where N is the number of expected datasets, p is the false positive rate of the Bloom filter, and L is the number of unique k-mers in the query.

A BIGSI was constructed for the classic reference dataset that all k-mer indexing tools benchmark against, the human RNA-seq dataset [291]. For BIGSI, the index size is 144 GB with a query time of 69 seconds. In comparison, SBT has an index size of 200 GB and a query time of 238 seconds.

4.2.7 COMPACT BIT-SLICED SIGNATURE INDEX

COmpact Bit-sliced Signature Index (COBS) [303] represents a significant advancement in k-mer indexing. It merges the q-gram (a q-gram is a contiguous sequence of q symbols within a text) indexing approach with Bloom filters to effectively index k-mers while reducing the k-mer space size. This novel combination is like a variant of signature files from the field of information retrieval, enabling rapid search for text data within a highly compact data structure, reducing disk access.

Unlike its predecessor, BIGSI, COBS is designed to accommodate samples with varying numbers of k-mers, addressing the challenge of over-scaling filters for smaller samples. While a BIGSI index requires all Bloom filters to be of the same size, resulting in denser bit vectors for larger samples and sparser for smaller ones due to the dependence on the number of q-gram terms or k-mers in a sample, COBS overcomes this limitation by creating multiple matrices corresponding to different filter sizes, thus optimizing space utilization. This crossover between an inverted index and Bloom filters in COBS offers scalability to larger document sets without requiring the complete index in RAM. COBS adjusts the size of each Bloom filter bit array to the document it indexes, maintaining a constant false positive rate. Despite potential concerns about query efficiency from querying multiple arrays, COBS implements SIMD instructions to enhance construction and query performance compared to BIGSI while reducing the final index size, especially in cases where sample sizes vary.

In its benchmarks, COBS outperforms other k-mer indexing tools in construction and query time. It displays the fastest construction time, significantly exceeding other tools such as ClaBS (BIGSI), Mantis,

SeqOthello, and AllSome SBT. In index construction, COBS exceeds its competitors by significantly. For instance, it takes only 43 seconds to construct an index from 1,000 documents, whereas ClaBS, Mantis, SeqOthello, and AllSome SBT lag considerably. COBS demonstrates superior speed, 2.3 times faster than ClaBS, 30 times faster than Mantis, 59 times faster than SeqOthello, and 83 times faster than AllSome SBT. Moreover, COBS highlights efficiency in the processing time compared to BIGSI, Mantis, and SeqOthello. Regarding query performance, COBS outperforms, requiring only 114 seconds to execute queries. In contrast, ClaBS takes 154 seconds, and Mantis and SeqOthello are even slower.

4.2.8 DYNAMIC SEARCHABLE PARALLEL COMPRESSED INDEX

The Dynamic seaRchablE pArallel coMpressed index (DREAM) [304] framework addresses the limitations of traditional mapping-based methods for indexing large genomic databases. As the size of reference databases used in genomic research has grown exponentially, traditional methods like FM-indexing [79] have struggled to keep pace, particularly with databases exceeding 10 GB. These methods require substantial time and computational resources to build and lack the flexibility needed for frequent updates, which is essential in fields like metagenomics, where reference sequences are continually changing.

The DREAM framework (Figure 4.8) represents an advanced indexing system that is developed to handle reads from Illumina and large-scale genomic databases. It introduces a dynamic, distributed approach to indexing that allows for efficient construction and fast search times. It begins with a set of database sequences and sequencing reads, which are split into smaller clusters or bins of similar sequences. Each cluster is indexed with a method of choice appropriate for the approximate search method, creating sub-indices. The framework has two main layers: the dynamic operation distributor and the approximate search distributor. The dynamic operation distributor manages updates by determining which sub-indices require modification and executing these updates in parallel. Meanwhile, the approximate search distributor determines which sub-indices to search, conducts these searches in parallel, and then consolidates the results. The effectiveness of this approach is enhanced by placing similar sequences into the same sub-index, which also aids in

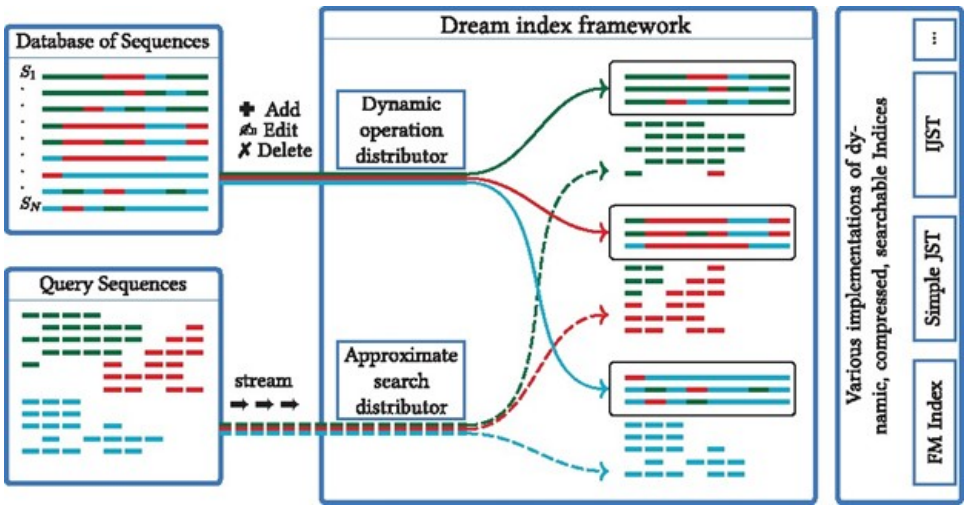

Figure 4.8: *(Source: (Dadi et al., 2018 [304])) Overview of the DREAM index framework.*

compression, although the specifics of compression were not discussed. The sub-indices use FM indices for fast approximate querying and can be dynamically rebuilt when necessary. The framework leverages the *k*-mer counting lemma ([304]) and a new Interleaved Bloom Filter (IBF) type for managing dynamic operations and approximate searches. The binning of sequences utilizes TaxSBP [305], a clustering method based on the NCBI Taxonomy database [306, 307], to group similar sequences. The IBF combines multiple Bloom filters into a single structure. Conventional BFs have limitations in storing binning bit vectors directly since they only handle set membership queries. The IBF interleaves multiple Bloom filters—one for each bin—into a unified Bloom filter. Each bit in the IBF represents a sub-bit vector for one bin, allowing simultaneous management of all bins. This interleaving technique improves query speed by making data retrieval more cache-friendly. In the context of the DREAM framework, the IBF helps determine which bins to search based on *k*-mer presence and updates these bins as the database evolves. Additionally, the Yara read mapper [308] is integrated into the DREAM framework, leading to the development of DREAM-Yara. This tool enhances Yara's capabilities by managing read mapping in batches, using the IBF to identify relevant bins, and consolidating results into a single output file.

DREAM-Yara was benchmarked using the set of archaeal and bacterial complete genome sequences retrieved from the NCBI database. The dataset comprised 15,250 sequences representing 2991 species, summing

up to a total of 31.34 Gbp. DREAM-Yara completed the indexing in 1 hour and 7 minutes for the dataset partitioned into 1024 bins, making it approximately nine times faster than Bowtie2 [80] and 26 times faster than standard Yara. Memory consumption was also lower, with DREAM-Yara using 16.15 GB of memory—62% less than BWA [76], the next most efficient method in terms of memory usage. DREAM-Yara's advantage extends to index updating as well; it only required 7 minutes to update the indices for the 155 new and one removed *E. coli* sequences, demonstrating an improvement over tools like standard Yara, BWA, and GEM [309], which would need to rebuild the entire index, taking up to a day.

4.2.9 KMTRICKS

Kmtricks [310], was designed to address the challenges encountered in indexing sequences generated by short-read technologies. Traditional indexing methods often struggle with the uncertainty of eliminating low-abundance *k*-mers while retaining erroneous ones, limiting their applicability in metagenomics and RNA-seq data analysis, especially in the absence of extensive reference genomes. Kmtricks emerged as a solution to optimize accuracy and efficiency in *k*-mer indexing by integrating *k*-mer counting and Bloom filter construction. Through techniques such as rescuing low-abundance *k*-mers, employing hash counting for simultaneous construction, and leveraging matrix transposition techniques, kmtricks offers a joint multi-sample *k*-mer indexing, addressing the shortcomings of traditional methods.

Kmtricks (Figure 4.9) workflow includes stages, such as partitioning, counting, and merging, to optimize the construction of *k*-mer matrices or Bloom filters from input sequencing data.

The partitioning strategy is based on minimizers to count *k*-mers and construct super-*k*-mers. Minimizers are the smallest subsequences within a sequence, while super-*k*-mers are sequences of *k*-mers that all share the same minimizer. The partitioning scheme ensures a balanced distribution of *k*-mers across partitions. This process involves sorting all possible minimizers and dividing them into partitions with roughly equal numbers of *k*-mers. In the counting stage, the super-*k*-mers for each sample are computed and written into the corresponding partitions

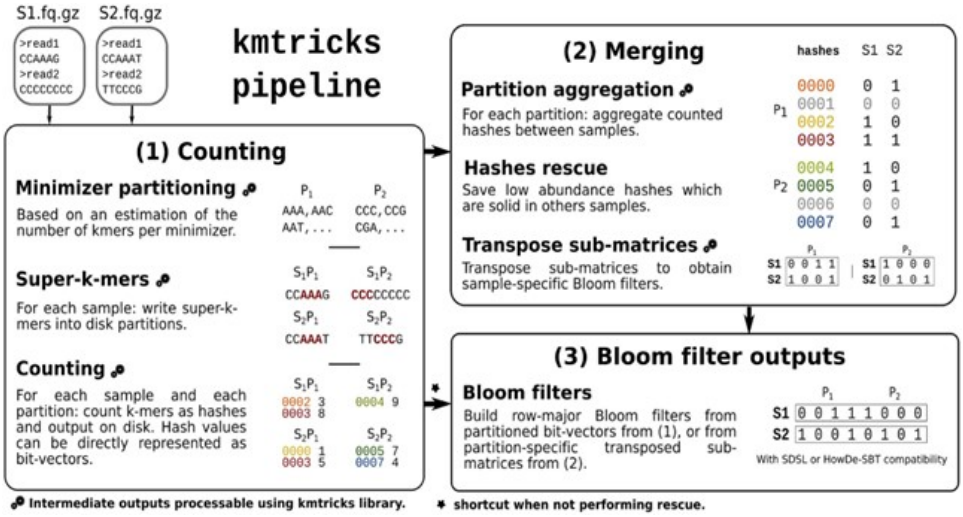

Figure 4.9: *(Source: (Lemane et al., 2022 [310])) An overview of the kmtricks pipeline.*

on the disk. These super-*k*-mers are used to deduplicate *k*-mer hash values and determine the abundance of each distinct hash value within each partition. During merging, hash value partitions combine to form Bloom filters in the rescue phase. In this process, the row count vectors are transformed into binary representations, and missing hash values result in the appending of empty bit-vectors to the matrix. After the merging step, the color-aggregative matrices are converted into *k*-mer aggregative representation through a bit-matrix transposition.

Kmtricks was benchmarked by conducting a joint *k*-mer counting and Bloom filter construction for a Tara Oceans dataset consisting of a 6.5 terabase metagenomics collection using under 50 GB of memory and 38 hours of construction time. It compared against the best alternatives by at least 3.8 times in speed. However, it required two times more storage compared with other tools.

Despite its strengths, kmtricks requires additional storage, posing challenges for resource-constrained environments. Furthermore, its reliance on third-party tools like HowDe-SBT for index construction may limit its applicability.

4.3 INTRODUCTION TO METAPROFI

In the last few years, several tools utilizing Bloom filter data structures for storing and querying large sequence datasets became available: SBT [291], SSBT [294], AllSome SBT [295], HowDe-SBT [297], BIGSI [302],

COBS [303], DREAM-Yara [304], kmtricks [310], and others. Other tools utilize other variants of probabilistic data structures: Squeakr [301] in combination with Mantis [298] and BCALM2 [311] in combination with REINDEER [312]. Custom indexes are built either from the raw sequencing data or the data from curated databases, and later, the sequences of interest are queried against these custom indexes to find which samples in the index contain the query sequence.

This section introduces two variants of MetaProFi, MetaProFi-1 [15] and MetaProFi-2. MetaProFi is a first-of-its-kind tool for indexing amino acid sequences that also supports nucleotide sequence indexing. Due to their numerous advantages and simplicity, MetaProFi is designed to use Bloom filters (BFs) as the underlying data structure. As described earlier, the Bloom filter (Figure 4.1) is a probabilistic set-membership data structure that stores the presence or absence of items/elements in a bit vector and can be queried for presence or absence.

MetaProFi (refers to both variants) combines the power of a variant of BF data structure, which we call packed Bloom filter (Chapter 4.4.2), with data chunking and compression to construct the BF matrix efficiently to index all the observed k-mers (presence/absence) for fast queries with reduced memory, storage, and runtime requirements.

4.3.1 NOVEL FEATURES

The novel features of MetaProFi include (1) indexing support for both nucleotide and amino acid sequences; (2) a possibility for querying amino acid sequence index using nucleotide sequences; (3) a seamless update (possible only in MetaProFi-1) of previously built indexes with new data/samples; (4) considerable storage reduction compared to the state-of-the-art tools.

4.3.2 APPLICATIONS AND PRACTICAL USAGE EXAMPLES

MetaProFi serves various applications: One can index all the sequences available in UniProtKB in a protein-based index. A possible application of such an index would be a fast alignment-free sequence search that can be used, for example, to find resistance-associated genes directly from metagenomics data. MetaProFi allows exact query search expecting

every *k*-mer in the query sequence to be present and also supports approximate search using a threshold (*T*) when only the fraction of *k*-mers larger than *T* have to be found. As a proof of concept, we have constructed a MetaProFi-1 index for UniProtKB bacterial sequences (amino acids) on two different levels (organism-level and sequence-level), a MetaProFi (both versions of the tool) index for the Tara Oceans dataset containing nucleotide metagenomic sequencing data collected across all oceans. Additionally, we used the dataset described in HowDe-SBT, which consists of 2585 human RNA-seq experiment results comprising blood, brain, and breast samples to demonstrate the storage, memory, run time, scalability, and query performance.

4.4 METAPROFI IMPLEMENTATION

MetaProFi (Figures 4.10, 4.11) is developed in Python and is parallelized to take advantage of all the CPU cores available in today's modern computing systems to achieve the best performance. MetaProFi accepts FASTA and FASTQ formats and uses pyfastx [313] Python library to process the files. The implementation is available at `https://github.com/kalininalab/metaprofi`.

For the downstream analysis of variants, we developed an API to integrate MetaProFi with StructMAn [314] (Chapter 5.2.1), providing efficient variant identification and structural annotation within a unified environment. Structural annotation refers to mapping the identified variants onto the 3D structure of proteins, to aid in understanding how these mutations might affect protein function and interactions. This is particularly useful in differentiating between benign and pathogenic mutations and identifying mutations linked to drug resistance. This API makes it easy to set up and configure MetaProFi and StructMAn tools together in one a single computational environment thereby simplifying their operations and compatibility. Once variants are identified, the API coordinates the transfer of these variants to StructMAn for structural annotation and functional impact analysis. The API manages all aspects of this process, including input management, error handling, and output generation, providing a robust and automated workflow for comprehensive analysis.

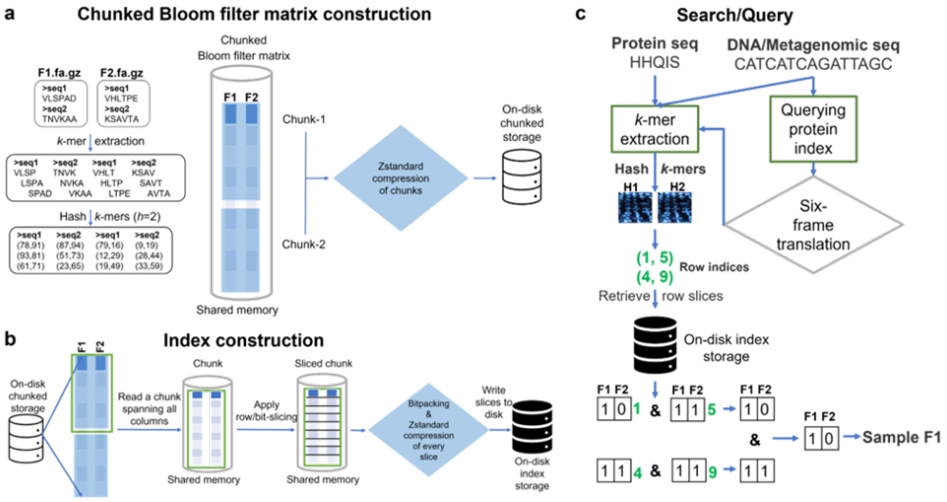

Figure 4.10: *Overview of MetaProFi-1 pipeline: (a) Chunked Bloom filter matrix construction, (b) Index construction, and (c) Search/query pipeline.*

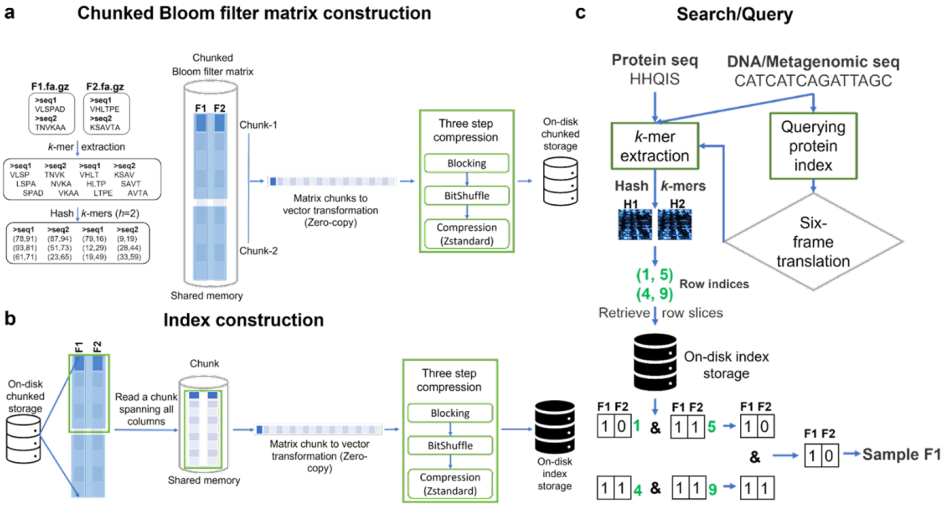

Figure 4.11: *Overview of MetaProFi-2 pipeline: (a) Chunked Bloom filter matrix construction, (b) Index construction, and (c) Search/query pipeline.*

4.4.1 EFFICIENT HASHING WITH MURMURHASH2 IN METAPROFI

MetaProFi implements a custom version of the MurmurHash2 algorithm (https://github.com/aappleby/smhasher/), a high-performance, non-cryptographic hash function designed to generate uniformly distributed hash values from arbitrary input data. It is known for its simplicity, speed, low collision rate, and good distribution. MurmurHash2 provides efficient and balanced data distribution across hash buckets. The algorithm employs bitwise operations to produce the final hash value, including shifts, XORs, and multiplications with constants. These operations ensure that even small changes in the input result in significant changes in the hash output, thus reducing the likelihood of collisions and providing a fast and efficient hashing mechanism [315–317].

4.4.2 CONSTRUCTION OF CHUNKED BLOOM FILTER MATRIX

MetaProFi builds BFs in the form of a matrix directly, unlike other tools that construct individual BFs for each sample first and then construct a matrix (and/or other forms) for indexing (Figures 4.10A and 4.11A). The rows in the matrix represent hash indexes, while each column represents a BF of length m of a sample. Since we cannot construct large matrices in memory, MetaProFi employs carefully crafted chunk size calculations to ensure efficient memory utilization and optimal performance during the BF matrix and index construction (Chapter 4.4.3). The chunk size is determined based on the user-defined configuration, particularly the allocated memory limit (M_{max}), the BF size m, and the number of input samples (Y). MetaProFi splits the number of samples into N batches of small samples, where N is estimated to fit within the user-defined maximum memory usage threshold.

Given that MetaProFi was implemented in Python, which lacks a native bit datatype, in the matrix, MetaProFi utilizes an 8-bit unsigned integer (UINT8) data type to store bits for eight k-mers by applying bit manipulations to a UINT8 integer. This effectively packs the BF (called a packed BF) and reduces the memory footprint by eight.

The total memory required for storing the BFs for all samples is calculated as,

$$\text{TotalBytes} = \text{PB} \times Y \tag{4.4.1}$$

$$\text{PB} = \left\lfloor \frac{m + Z}{8} \right\rfloor \tag{4.4.2}$$

Where *PB* represents the packed bytes per BF, Z is the number of zeros added to align the bit count to the nearest byte boundary.

To determine the number of chunks needed, MetaProFi divides the total required bytes by a fraction (85%) of the maximum memory allowed by the user,

$$N = \frac{\text{TotalBytes}}{0.85 \times M_{\text{max}}} \tag{4.4.3}$$

this leaves a margin to ensure that operations remain within the allocated memory. The chunk size in terms of the number of samples is then calculated as,

$$C_{\text{samples}} = \left\lfloor \frac{Y}{N} \right\rfloor \tag{4.4.4}$$

and the chunk size in terms of the number of rows is calculated as,

$$C_{\text{rows}} = \left\lfloor \frac{\text{PB}}{8 \times N} \right\rfloor \tag{4.4.5}$$

Here, dividing by eight accounts for the Bloom filter's bit-packing into a UINT8 integer. These carefully computed chunk sizes allow MetaProFi to process a larger number of samples per iteration while staying within memory limits, thus optimizing both memory utilization and processing efficiency.

Once all *k*-mers in a batch of samples are hashed and the respective bits are flipped in the BF matrix, MetaProFi applies a structured approach to efficiently manage and store the matrix. This is achieved using Zarr [318], an open-source library for storing and processing large, chunked arrays. It is combined with compression techniques such as the Zstandard algorithm (`https://github.com/facebook/zstd`) to reduce storage

requirements for large-scale datasets. MetaProFi's chunking strategy organizes the BF matrix into manageable chunks based on the previously calculated chunk sizes ($C_{samples}$ is the number of columns in the matrix, and C_{rows} is the number of rows in the matrix). Each chunk represents a subset of the matrix, allowing for efficient processing and retrieval. Zarr facilitates this process by supporting various storage backends, including local file systems and cloud storage, allowing for independent access or updates to each chunk. This integration with Zarr optimizes memory usage, enhances processing speed, and ensures effective management of large-scale sequence datasets, perfectly aligning with MetaProFi's objectives.

In MetaProFi-1, the BFs from all batch samples are divided into chunks based on the calculated chunk sizes $C_{samples}$ and C_{rows}. Each chunk is then compressed and written to the disk. The chunking strategy ensures that a portion of the full BF matrix, corresponding to the number of samples and rows defined by $C_{samples}$ and C_{rows}, can be loaded into memory for processing.

Whereas MetaProFi-2 applies the following four-step compression technique to each chunk (C) in the matrix: (1) each chunk is vector transformed (flattened), (2) the vector is split into blocks that can fit into the CPU cache, (3) a Bitshuffle [319] algorithm is applied to the blocks, and (4) the Bitshuffled blocks are then compressed and the compressed blocks are then stitched back to form a single vector. This compressed vectored chunk is then written to the disk. MetaProFi-2 leverages the Blosc (`https://github.com/Blosc/c-blosc`) tool to perform these operations.

By applying compression to each chunk, MetaProFi-1 achieves a significantly better compression ratio than when compressing individual Bloom filters, and through this, MetaProFi-1 offers a significant storage reduction. MetaProFi-2 further improves this by incorporating a sophisticated blocking and bit shuffling technique. Blocks in MetaProFi-2 are sized to fit into the CPU cache, a small high-speed memory area on the processor that temporarily holds frequently accessed data. By ensuring smaller block sizes, the algorithm minimizes the need for slower main memory accesses, enhancing processing speed and efficiency. The Bitshuffle algorithm rearranges the bits into a matrix with dimensions of the number of elements by the size of each element (in bits) and

then performs a transpose [319] operation, which aligns similar bits into longer contiguous sequences (run lengths) that can be compressed better. Combining optimal block size with bit shuffling leads to more effective data compression, thereby reducing the storage to a greater extent than any of the *k*-mer indexing tools.

This process is repeated for all N sample batches. Using the Zarr library to store the chunks on disk ensures that the chunks from all samples that correspond to the same set of Bloom filter rows are stored in such a way that they can be assessed and loaded to memory simultaneously. The whole bit vector corresponding to a single hash value in all Y samples can be extracted.

MetaProFi utilizes POSIX shared memory as the matrix backend, enabling efficient, concurrent access to the matrix through multiple processes. POSIX shared memory allows different processes on the same machine to share a common memory space, providing a zero-copy data transfer. This eliminates the need for redundant data copying between processes, significantly reducing the overhead associated with inter-process communication and accelerating the Bloom filter construction time. The primary advantage of using a POSIX shared memory in MetaProFi is its ability to enable multiple processes to work simultaneously on different columns (each column in the BF matrix is a sample) of the BF matrix. This parallelization substantially reduces the time required to construct large BF matrices, as each process can directly access and modify the shared matrix. This combination of shared memory optimization, chunking strategy, and the Zarr library enhances MetaProFi's speed and scalability and optimizes memory usage.

4.4.3 INDEX CONSTRUCTION FOR THE BLOOM FILTER MATRIX

Since MetaProFi's BF matrix is stored in batches and chunks, direct queries for a large number of *k*-mers need to enumerate all chunks and likely will be slow. So, we build a dedicated, efficient index structure (Figures 4.10B and 4.11B) for the BF matrix such that querying every *k*-mer has a constant time cost [302] independent of the number of batches.

The indexing data structure of MetaProFi-1 is an array of size m (size of the Bloom filter), where each cell in the array corresponds to a row (or, equivalently, a bitslice [302]) across all samples (columns) from the BF matrix. Whereas MetaProFi-2 constructs a type of sparse index to store the bitslices where keys are chunk numbers, and the values are the compressed vectors (more than one bitslice) and stores them in a key-value database called LMDB (Lightning Memory-Mapped Database) (`https://lmdb.readthedocs.io/en/release/`). LMDB provides several advantages, including efficient memory mapping, which allows the data to be accessed directly from the disk without loading it fully into the memory. This memory-mapped approach significantly reduces the memory overhead and enables faster access times by mapping disk files into the process's address space. Moreover, LMDB offers transaction support, ensuring data integrity and reliability during concurrent read and write operations. By leveraging these features, MetaProFi-2 enhances its indexing efficiency and scalability, making it robust when handling extensive data collections. The combination also speeds up data retrieval and storage operations and efficiently manages large-scale datasets without compromising performance or reliability.

To construct the index, MetaProFi re-creates a POSIX shared memory matrix of size $X * Y$, where Y is equal to the total number of samples in the BF matrix and $X = C_{rows}$ is the number of rows that can be read into the memory without crossing the maximum memory limit (M_{max}) set in the user configuration file. MetaProFi reads X rows from each chunk spanning all N sample batches on the fly, unpacks the UINT8 packed Bloom filter to individual bits, and writes the several unpacked bits of X rows and $C_{samples}$ corresponding to a batch to the shared memory matrix in parallel.

Once again, to benefit from data chunking and compression, MetaProFi-1 applies this technique to these X rows and distributes the compressed chunks to multiple processes, which are then written to the disk. Meanwhile, MetaProFi-2 chunks these X rows, applying the four-step compression technique (Chapter 4.4.2) to each chunk and then writing them to the key-value database.

MetaProFi repeats the procedure until all rows from the BF matrix are indexed. MetaProFi-1 supports updating the index to add new samples, while MetaProFi-2 does not offer this feature yet.

4.4.4 FAQINDEXING FOR FASTA/FASTQ FILES

MetaProFi can build the BF matrix using a collection of FASTA/FASTQ files where each file is treated as a single sample or from a single FASTA/FASTQ file, where every sequence is treated as an individual sample. In the latter case, we introduce a dedicated indexing data structure to accelerate the MetaProFi BF matrix construction, referred to hereafter as FAQIndex (FASTA/FASTQ Index). Inspired by pyfastx [313], we implemented an LMDB-based indexing tool for compressed and uncompressed FASTA/FASTQ files, leveraging LMDB as the storage backend due to the advantages described earlier. FAQIndex is designed to contain six columns: the sequence number, the name of the sequence, the sequence start offset, the byte length of the sequence, and the number of bases in the sequence. This indexing approach significantly enhances the efficiency of sequence data processing by enabling rapid access to specific sequences without the need to scan through entire files. By knowing each sequence's exact location and length in the file, MetaProFi can quickly retrieve and process the sequence data, minimizing I/O operations and reducing the time required to build the BF matrix.

A key advantage of FAQIndex over other FASTA/FASTQ indexing tools is its support for concurrent access. FAQIndex allows multiple processes to access and manipulate the index simultaneously. LMDB's architecture makes this concurrency possible, ensuring smooth and efficient parallel processing. As a result, MetaProFi can parallelize the construction of the BF matrix across multiple processes, leading to a significant reduction in overall processing time; this also optimizes memory usage by allowing MetaProFi to load only the data required for the current operation into memory rather than loading the entire file. This is advantageous where memory constraints can become a bottleneck. FAQIndex is also designed to be space efficient, as we do not store additional information, such as sequence type and read parameters, requiring less storage than Pyfastx's. Each row in FAQIndex is serialized and compressed, reducing storage requirements. This decreases storage (file size) and enhances access speed, as smaller indexes can be loaded and searched more quickly.

4.4.5 QUERYING/SEARCHING THE METAPROFI INDEX

MetaProFi accepts raw sequence and FASTA/FASTQ files as inputs for querying the index (Figures 4.10C and 4.11C). When a multi-sequence file is used for querying, MetaProFi automatically constructs a small FAQIndex of the file to distribute the query sequences to multiple cores/processes. MetaProFi collects the hashes of each k-mer from every sequence in parallel processes. The MetaProFi index is queried with these hash values, and bitslices corresponding to each k-mer are retrieved.

MetaProFi allows exact query search where every k-mer in the query sequence is expected to be present in a sample and also supports approximate search using a threshold (T) when only a fraction of k-mers larger than T have to be found. In addition to querying with an amino acid sequence against an index built using amino acid samples and a nucleotide sequence against a nucleotide index, MetaProFi allows querying an amino acid index using nucleotide sequences (e.g., metagenomic reads, contigs, or assembled genomes) directly. To this end, MetaProFi performs a six-frame translation of the nucleotide sequences, which means it translates each nucleotide sequence in all six possible reading frames (three frames in the forward direction and three frames in the reverse direction) to generate six different amino acid sequences. These six translated sequences are then used as queries to search the amino acid index. This approach is not expected to create false positive hits because, on average, a stop codon appears approximately every 21 codons in the five non-biological frames. The genetic code contains three stop codons out of 64 possible codons, meaning the probability of encountering a stop codon is $3/64$. Therefore, the expected average distance between stop codons is roughly $64/3 = 21.33$ codons. Hence, not more than ten consecutive k-mers can be matched in a spurious translation frame.

4.4.6 FALSE-POSITIVE RATE

Bloom filters belong to the class of probabilistic data structures with a zero false-negative rate, and they are prone to false positives by design. However, the false-positive rate of the Bloom filters can be controlled by increasing the size of the Bloom filter and the number of hash functions used. As discussed in Chapter 4.1.3.1 and in [302] one can also calculate

the false positive rate of the query, which depends on 1) the number of samples in a dataset, 2) the size of the *k*-mer, 3) the maximum number of acceptable false discoveries per query, and 4) the shortest length of the query sequence to be supported. Using these parameters, a false positive rate per query can be calculated. For example, the exact formulation for the false positive rate calculation is presented in Chapter 4.1.3.1 and in [302].

4.4.7 COMPUTING SETUP

Performance evaluations were done on a Dell server with the following configuration: AMD EPYC 7702 2.0 GHz CPU with 1.5 TB RAM, Intel SSD DC P4610 3.2 TB (2.9 TiB), and CentOS 7 operating system. MetaProFi was allocated 64 cores and 60 GiB RAM for all its experiments, and the same was done for other tools wherever possible during benchmarking. All input files were stored on a non-RAID NVMe NFS file system, and outputs were stored on the Intel SSD DC P4610 3.2 TB disk.

4.5 BENCHMARKING DATASETS AND PARAMETERS

4.5.1 UNIPROTKB DATASET

Two types of MetaProFi-1 indexes, one at the organism level and one at the sequence level were constructed for all bacterial sequences in the UniProtKB (Swiss-Prot and TrEMBL) database downloaded in July 2021, which has a total size of 64 GiB. The entire Swiss-Prot and TrEMBL datasets were downloaded from UniProt's FTP site, and the accession IDs for all the bacterial sequences were downloaded by performing a search with *"taxonomy:bacteria"* on UniProt's search interface.

Three parameters define the architecture of Bloom filters in MetaProFi: m, h, and k, where m is Bloom filter size, h is the number of hash functions to be applied on every k-mer, and k is the size of the k-mer. In MetaProFi-1, for organism-level indexing, we used the following parameters: $m = 600,000,000$; $h = 2$; $k = 11$. With these parameters, the false positive rate was 0.47, while the false positive rate per query is 10^{-6} if the query sequence size is a minimum of 35 characters. We used

the same Bloom filter parameters with one change, $m = 600,000$, for sequence-level indexing of the UniProtKB bacterial dataset. With these parameters, the false positive rate was 0.0156, and the per query false positive rate is 10^{-6} if the query sequence size is of a minimum of 34 characters.

4.5.2 TARA OCEANS DATASET

MetaProFi was mainly developed to fill the technology gap of amino acid sequence indexing, but nucleotide sequence indexing support was also added for benchmarking purposes. With this feature, we downloaded 4 TiB of compressed Tara Oceans dataset (study accession: PRJEB1787) from the ENA archive consisting of 249 samples containing 495 FASTQ files. For MetaProFi indexing of the Tara Oceans dataset, we used the following Bloom filter parameters: $m = 40,000,000,000$; $h = 1$; $k = 31$. With these parameters, the false positive rate was 0.3782, and the false positive rate per query is 10^{-6} if the query sequence size is of a minimum of 50 characters. Since we wanted to benchmark MetaProFi's performance with other tools that do not allow changing the number of hash functions, we set MetaProFi to use only one.

We compared MetaProFi's performance with kmtricks (v1.1.1), and kmtricks was run using the same Bloom filter parameters as above and without filtering k-mers that appear only once. Further, kmtricks were run in the "hash:bft:bin" mode, which only performs the BF matrix construction instead of the k-mer counting. For a fair comparison, we chose other parameters (number of cores, k-mer length) to match those of MetaProFi. For kmtricks in combination with HowDe-SBT, we used only 1% of bits to be considered from all Bloom filters during clustering and indexing since the value recommended by the HowDe-SBT tutorial (`https://github.com/medvedevgroup/HowDe-SBT/tree/master/tutorial`) lead to prohibitively long runtimes.

We randomly selected 1000 reads from the 495 FASTQ files of the Tara Oceans dataset to evaluate query performance and used them for querying.

4.5.3 HUMAN RNA-SEQ DATASET

For benchmarking, we also used a human RNA-seq dataset consisting of 2585 samples (2.7 TiB) that were also used in [297]. We downloaded this dataset from the SRA using the parallel-fastq-dump (https://github.com/rvalieris/parallel-fastq-dump) tool; accession numbers were obtained from [297].

We divided this experiment into two sets. First, a subset of 650 samples (referred to as RNA-seq-mini hereafter) was randomly selected for building a small index to compare the performance of several tools: HowDe-SBT (v2.00.0220191014), kmtricks (v1.1.1), COBS (v0.1.2), Squeakr (v1.0) in combination with Mantis (v0.2.0), and MetaProFi. Second, all 2585 samples (referred to as RNA-seq hereafter) were indexed using both COBS (v0.1.2) and MetaProFi.

We chose the Bloom filter parameters for the RNA-seq-mini dataset: $m = 2,000,000,000$; $h = 1$; $k = 21$. With these parameters, the false positive rate was 0.09, and the false positive rate per query was 10^{-5} with a minimum query size of 31 characters. To level the comparison and consistency between all tools, we first constructed compacted De Bruijn Graphs for all 650 samples using BCALM2 (v2.2.3) [311] while removing all k-mers that appear only once and then used this as input to all the tools for the benchmark. We built Squeakr input files for Mantis and removed all k-mers that appear only once. It must be noted that MetaProFi does not require these preprocessing steps. While executing all tools, we ensured that none of them repeated the removal of k-mers that appeared only once step. Also, we ran kmtricks in the *"hash:bft:bin"* mode, which will only perform the BF matrix construction instead of the k-mer counting. For a fair comparison, we chose other parameters (number of cores, k-mer length) to match those of MetaProFi. For HowDe-SBT and kmtricks in combination with HowDe-SBT, we used only 1% of bits to be considered from all Bloom filters during clustering and indexing since the value recommended by the HowDe-SBT tutorial (https://github.com/medvedevgroup/HowDe-SBT/tree/master/tutorial) lead to prohibitively long runtimes.

For the RNA-seq dataset, we obtained the Bloom filter parameters from [312]. Bloom filter parameters were the following: $m = 2,000,000,000$; h

= 1; k = 21. This dataset was used as it is without applying any k-mer filtering.

We downloaded a FASTA file comprising 70,866 transcripts to evaluate query performance, following [312]. We then extracted the first 1000 transcripts using pyfastx to query all tools' RNA-seq and RNA-seq-mini indexes. RAM utilization was monitored through the Linux command-line utility *atop* (via the command *atop -mp*).

4.6 COMPARATIVE BENCHMARKS OF METAPROFI AND OTHER TOOLS

MetaProFi allows the indexing of large numbers of samples/datasets. MetaProFi (Figures 4.10 and 4.11) combines the power of a probabilistic data structure with data chunking and compression to store large Bloom filters and to create indexes for fast querying. The key methodological novelty of MetaProFi is its ability to index amino acid sequences and enable the querying of amino acid sequence index using nucleotide sequences. This allows disregarding synonymous mutations and focusing directly on sequence variants that impact the protein sequence and, hence, may impact the corresponding protein functions. Moreover, more efficient exact sequence searches are also possible for non-exactly matching strains that contain only silent mutations. Additionally, since protein sequence homology is detectable across longer evolutionary distances, homologous sequences can be detected on the level where nucleotide-level similarity fails.

To evaluate MetaProFi's performance, we used UniProtKB, Tara Oceans, human RNA-seq, and RNA-seq-mini datasets (Chapter 4.5) for the index construction. For UniProtKB, two indexes were created: one at the organism level and the other at the sequence level. Since no other tool is available to perform amino acid k-mer indexing, we added support for nucleotide indexing to MetaProFi to enable comparison against other tools, and we indexed the Tara Oceans and the RNA-seq datasets for benchmarking purposes.

4.6.1 UNIPROTKB DATASET INDEXING

For UniProtKB organism-level indexing, the dataset was constructed by extracting individual bacterial sequences using their accession IDs with criteria on the minimum length of the sequence (k = 11) and then grouping them by their organism's name (OS field value in the FASTA header) to obtain 100,384 uncompressed individual fasta files containing a total of 46,511,863,142 k-mers. MetaProFi-1 constructs the BF matrix in 38.05 min and the index in 971 min using under 60 GiB of RAM and 135 GiB of disk space (Table 4.1). Storage size is directly proportional to the size of the Bloom filter and the number of samples in the dataset: if MetaProF-1 had used a regular Bloom filter, it would require 6.85 TiB (size of the Bloom filter times number of fasta files/samples, i.e., $600000000 * 100384 = 60230400000000$ bits, which is equal to 6.85 TiB) disk space for storing the uncompressed BF matrix. With MetaProFi's optimizations and techniques, we provide a 50-fold compression. We can construct Bloom filter matrices for a large number of datasets or use very large Bloom filters that have a low false-positive rate while still storing them efficiently.

	Time (min)	RAM (GiB)	CPU cores	Disk (GiB)
Organism-level				
Bloom filter matrix	38.05	< 60	64	139
MetaProFi-1 index	971	< 60	64	135
Sequence-level				
Bloom filter matrix	65.38	< 60	64	232
MetaProFi-1 index	1357	< 60	64	210

Table 4.1: *MetaProFi-1 indexing results for UniProtKB bacterial dataset.*

To demonstrate MetaProFi-1's scalability, we used all the bacterial sequences that were extracted from the UniProtKB dataset, 334,984 sequences from Swiss-Prot, and 151,450,171 sequences from TrEMBL. We monitored the construction of the FAQIndex (Chapter 4.4.4) for the compressed UniProtKB bacterial input dataset of size 32 GiB during the BF matrix build step. MetaProFi's FAQIndexing tool took 17 minutes to construct the FAQIndex and used 11 GiB of storage and a maximum

of 1 GiB of RAM. Using LMDB as the underlying database reduces the RAM consumption as we do not retain data in memory and write the data for every sequence to the disk as soon as they are populated. Using the FAQIndex of the UniProtKB bacterial input dataset (151,785,155 sequences), we created a sequence-level MetaProFi-1 index. The BF matrix and index construction time are comparable with the organism-level case. The disk requirements are twice as large, and RAM consumption is comparable (Table 4.1). These results show that MetaProFi-1 is scalable to hundreds of millions of samples.

4.6.2 RNA-SEQ-MINI DATASET INDEXING

We created a subset of 650 samples out of 2585 samples of the RNA-seq dataset to benchmark MetaProFi's performance with other tools such as HowDe-SBT, kmtricks in combination with HowDe-SBT, COBS, and Squeakr in combination with Mantis. We first built the compacted De Bruijn Graphs for all 650 samples (while removing k-mers that were present only once) and then used them as the input (Chapter 4.5.3). From the results (Table 4.2), we can see that COBS has the best total runtime, followed by MetaProFi, whereas the total disk consumption of MetaProFi is the smallest.

Tool	RAM (GiB)	CPU cores	Disk BF (GiB)	Disk index (GiB)	Disk total (GiB)	Time BF (min)	Time index (min)	Total time (min)
HowDe-SBT	2.4	64	152	4.4	168.4	51.94	93.1	145.1
Kmtricks + HowDe-SBT	286.39 + 4	64	156	4.4	308 + 12	54.49	383.9	438.39
MetaProFi-1	12	64	8.2	9.4	17.6	4.37	20.42	24.79
MetaProFi-2	12	64	5.94	5.36	11.3	4.05	14.84	18.89
COBS	12	64	—	51	54	—	8.59	8.59
Squeakr + MANTIS	7.3 + 49	64	28	14	42	145.22	33.27	178.49

Table 4.2: *RNA-seq-mini dataset indexing benchmark comparisons, BF: Bloom filter, Disk total: total storage used for BF, index, and intermediate files, Time total: total time for constructing BF and index, — = N/A*

We also benchmarked the query performance of all these tools, for which we downloaded a FASTA file from RefSeq [320] comprising 70,866

human transcripts from RefSeq, as reported in [312]. We extracted the first 1000 transcripts and used them for querying. The results show that HowDe-SBT is faster when performing exact querying (T = 100%), whereas MetaProFi-1 is equally fast in both exact and approximate (T = 75%) searches and requires a very low amount of memory (Table 4.3).

Tool	RAM (GiB)	CPU cores	Time (s) (T = 100%)	Time (s) (T = 75%)
HowDe-SBT	0.61	–	22	558
kmtricks + HowDe-SBT	0.64	20	2952	2957
MetaProFi-1	1.9	20	29	33
MetaProFi-2	1.8	20	42	43
COBS	25.1	20	234	228
Squeakr + MANTIS	14	–	37	–

Table 4.3: *RNA-seq-mini query performance benchmark of* 1000 *transcripts.*

4.6.3 RNA-SEQ DATASET INDEXING

We built a complete index with all the samples (2585 samples; k = 21; 6,432,932,578,661 k-mers) from the human RNA-seq experiments obtained from the SRA using MetaProFi and COBS. MetaProFi takes a bit longer than COBS but requires much less storage (Table 4.4). We did not attempt to build the index with other tools as they were found to be prohibitively slow for the small RNA-seq-mini dataset.

Tool	RAM (GiB)	CPU cores	Disk BF (GiB)	Disk index (GiB)	Disk Total (GiB)	Time BF (min)	Time index (min)	Total Time (min)
MetaProFi-1	59	64	295	333	628	1108	127	1235
MetaProFi-2	59	64	272	296	568	1301	49	1350
COBS	69.4	64	N/A	935	996	N/A	1000	1000

Table 4.4: *RNA-seq dataset indexing benchmark comparisons.*

We then used this index to query 1000 transcripts (Table 4.5). MetaProFi was 6-7 times faster than COBS and required much less memory.

Tool	RAM (GiB)	CPU cores	Time (s) (T = 100%)	Time (s) (T = 75%)
MetaProFi-1	3.4	64	43	48
MetaProFi-2	4.2	64	54	61
COBS	92.5	64	290	290

Table 4.5: *RNA-seq query performance benchmark of* 1000 *transcripts.*

4.6.4 TARA OCEANS DATASET INDEXING

To compare MetaProFi with the state-of-the-art tools, we used kmtricks, a k-mer counting tool that allows building Bloom filters that can be utilized for constructing a k-mer index using a variant of HowDe-SBT implemented in its package. We applied both tools to the Tara Oceans dataset, which contains $3,431,551,187,218$ k-mers ($k = 31$).

During Bloom filter construction, kmtricks was 2-3 times faster than MetaProFi while consuming 2-3 times more storage than MetaProFi (Table 4.6). After 120 hrs, the kmtricks + HowDe-SBT index construction (1% of bits were considered from each filter) was terminated. We report only the numbers we observed until the termination without extrapolation (we assume that the computation might have taken several more days as only less than half of the Bloom filters were indexed at the time of termination). On the other hand, we can see that MetaProFi requires very little time to build an index. This shows that MetaProFi can index datasets containing trillions of k-mers in a reasonable amount of time yet only requiring low amounts of memory and storage.

Tool	RAM (GiB)	CPU cores	Disk BF (GiB)	Disk index (GiB)	Disk Total (GiB)	Time BF (min)	Time index (min)	Total Time (min)
MetaProFi-1	68	64	643	750	1393	2642	279	2921
MetaProFi-2	60	64	622	626	1248	2709	101	2810
kmtricks + HowDe-SBT	47	64	1228.8	$> 390^*$	$2344 + 390^*$	865	$> 7217^*$	$> 8082^*$

Table 4.6: *Tara Oceans dataset indexing benchmark comparisons of MetaProFi and kmtricks; BF: Bloom filter, Disk total is the total storage used for BF, index, and intermediate files, Time total: total time used to construct BF and index, *: terminated after* 120 *hrs, data reported as it is at the time of termination.*

To demonstrate MetaProFi's query performance, we randomly selected 1000 reads from the 495 FASTQ files of the Tara Oceans dataset used for constructing the index. These 1000 reads were queried against the Tara Oceans MetaProFi-1 and MetaProFi-2 index using exact search (T = 100%) and approximate search (T = 75%). The query run times were 164 seconds and 176 seconds for the exact search (T = 100%) and 166 seconds and 179 seconds for the approximate search (T = 75%). We observed a peak memory usage of 14.3 GiB for MetaProFi-1 and 7.1 GiB for MetaProFi-2. We could not compare the query results with the kmtricks + HowDe-SBT setup, as the index construction had to be terminated after 120 hrs.

Our benchmarking results show that MetaProFi reduces disk usage even for very large Bloom filters compared to state-of-the-art tools, constructs an index in little time, and performs better during querying.

4.7 DISCUSSION

MetaProFi, for the first time, presents a possibility to index protein sequences directly, which makes calling variants in coding sequences a much easier task. In addition, it features a mode of usage centered around nucleotide sequences, which makes it possible to compare it to other tools in the field. In these comparisons, both versions of MetaProFi demonstrated state-of-the-art performance with the best runtime/memory/storage ratio. MetaProFi was able to build k-mer indexes rapidly for multiple datasets of different sizes, demonstrating it can scale in any direction.

Nevertheless, the most crucial feature of MetaProFi, which makes it stand out among other tools, is that it can build indexes for amino acid sequences and enables querying of an amino acid index using nucleotide sequences. This approach was intended to store and query sequence data from metagenome samples, primarily bacterial metagenomes, as efficiently as possible. Storing protein data makes the search more flexible and offers many advantages, while the only disadvantage is that the information in non-coding regions is lost. In this scenario, we consider this a little loss since bacterial genomes contain comparatively little non-coding sequence, and many essential markers are detected at the protein level, e.g., markers of antibiotic resistance. Potential advantages

include, for example, the possibility of conducting swift and efficient searches with the k-mer presence threshold $T = 100\%$ for closely related but not identical DNA sequences that contain no missense mutations at the protein level. On the other hand, remote homologs can be detected in cases when the sequence similarity on the DNA level drops but is still detectable at the protein level. While other state-of-the-art tools do not typically offer the functionality to store amino acid-based indexes, the code change to allow it would not be a large one *per se*, although it would require numerous adjustments. However, the option to query an amino acid-based index with a nucleotide query is non-trivial and unique to MetaProFi.

Finally, in this work, we have developed a first-of-its-kind amino acid k-mer indexing tool with added support for indexing nucleotide sequences that efficiently builds indexes from tens of samples to hundreds of millions of samples with reduced memory, storage, and runtime compared to its predecessors. MetaProFi also addresses scaling problems in different contexts. Various optimizations such as shared memory utilization, memory-mapped files, new compression techniques, algorithms, and indexing methods were explored and implemented in MetaProFi. Through our proof-of-concept index construction for multiple datasets, we demonstrated that we could grow our index horizontally (samples) or vertically (Bloom filter size). Yet, it requires very little storage and memory without compromising index building and querying performance. MetaProFi can be further developed to support distributed computing infrastructure in addition to the current single system-specific deployment setup.

5

PERSPECTIVE

5.1 CONCLUSIONS

Drug resistance is a major problem, significantly affecting the efficacy of current therapies and often leading to relapse and treatment failure. The emergence of the acquired secondary drug resistance mutations, are a critical contributing factor in this challenge. Mutations can affect protein conformation and drug-binding sites, leading to therapy failure and relapse in diseases like cancer or infectious diseases. These mutations reduce drug efficacy by affecting essential protein functions such as ligand binding, conformational stability, and allosteric regulation, ultimately contributing to resistance against first- and second-line drugs. Understanding the molecular basis of drug resistance requires a multidimensional approach. The work performed in this thesis addresses the problem by employing two complementary strategies.

First, MD simulations study how specific mutations impact protein dynamics, structure, and conformational changes crucial for activation and ligand binding at the atomistic level. These simulations provide insights into how mutations contribute to drug resistance. Second, MetaProFi, a novel tool utilizing an ultra-fast chunked Bloom filter, efficiently addresses the significant computational challenges of analyzing vast sequencing datasets. MetaProFi facilitates the accurate identification of functionally relevant genetic variants by enabling comprehensive indexing and querying of both protein and nucleotide sequences. These approaches allow for a thorough analysis of drug resistance mechanisms, from identifying genetic variants to studying their impact on protein structure and function, eventually supporting the development of more effective therapeutics.

In Chapter 3, we employed a unified MD simulations methodology to study and understand the impact of mutations on drug resistance mechanisms. In the first case study, we focused on the KIT receptor tyrosine kinase, studying the impact of phosphorylation of Y823 on the

KIT protein 3D structure. Our simulations showed that phosphorylation at Y823 stabilizes the active conformation of KIT. This stabilization is acquired through the formation of H-bonds and enhanced interactions within the protein's activation loop, thereby shifting the dynamic equilibrium towards the active state, which is less responsive to inhibition by drugs targeting the inactive state of the protein. In comparison, the Y823D mutation, which introduces a negative charge at the same position, mimics the effects of phosphorylation by destabilizing the inactive conformation and promoting the active state. Our comparative analysis reveals that both the Y823D mutation and phosphorylation at Y823 promote the active conformation of KIT but through different mechanisms. Given the similarity of this mechanism to the mechanistic effects of phosphorylation and mutations at position Y393 in Abl kinase [224], where both destabilize the inactive conformation, our results suggest a potential analogous resistance mechanism in homologous kinases. In the second case study, we applied the same methodology to the NS3-Q80K variant in the NS3-4A protease of the hepatitis C virus. The Q80K mutation is associated with resistance to direct-acting antiviral agents and is prevalent in certain viral strains. Our simulations showed that the Q80K mutation destabilizes the protease structure, particularly in the N-terminal subdomain, reducing protein stability. This destabilization effect is compensated by epistatic amino acid substitutions at residues 91 and 174, which enhances the stability of the protein.

In Chapter 4, we present MetaProFi, a novel tool developed to address the computational challenges associated with analyzing vast sequencing datasets. Unlike traditional tools like BLAST, which struggle with scaling issues, MetaProFi offers a significant advancement in addressing these challenges. It introduces a novel approach by combining Bloom filters with advanced optimizations, including shared memory systems, efficient data chunking, and compression algorithms. MetaProFi uniquely allows for indexing nucleotide and amino acid sequences, enabling nucleotide queries against amino acid indexes. This feature is crucial for in-depth sequence analysis and identifying mutations. Better conservation of protein sequences across evolutionary distances makes MetaProFi particularly robust in detecting non-identical but closely related sequences and homologs. For example, MetaProFi can index comprehensive databases like UniProtKB, allowing direct querying of this

protein index using nucleotide reads or sequences from metagenomic or microbiome experiments. This alignment-free approach accelerates the identification of functionally relevant genetic variants. MetaProFi's capability to query non-perfectly identical sequences further adds to its flexibility and state-of-the-art performance, handling large-scale datasets with improved efficiency while maintaining an excellent balance between the usage of compute resources. This makes MetaProFi an important tool not only for handling large-scale datasets efficiently but also for identifying and characterizing mutations, thereby addressing complex biological questions.

5.2 OUTLOOK

MetaProFi can be extended to include the k-mer counts during indexing, which is widely used in genomic sequence analysis to identify patterns and detect genomic similarities and differences across datasets. It would also facilitate functional annotation and sequencing error correction. This feature would enhance MetaProFi's capabilities for comparative genomics and metagenomic analysis. Further, the FAQIndex from MetaProFi can be extracted as a standalone package to offer faster indexing for FASTA/FASTQ files with minimal storage requirements. It supports concurrent access, thereby optimizing data retrieval and processing speed. This feature would greatly benefit applications needing rapid sequence data access and processing, especially in resource-limited environments.

5.2.1 METAPROFI-STRUCTMAN INTEGRATION

StructMAn [314] is a tool developed for the structural annotation of non-synonymous single nucleotide variants (nsSNVs), which are mutations that alter amino acids in proteins that often lead to diseases. StructMAn provides insights into the functional impact of these genetic variants by incorporating the 3D structural context of proteins. The tool analyzes the spatial location of the amino acid residues affected by nsSNVs within the protein's 3D structure and takes into account its surrounding structural environment, including interactions with other proteins, nucleic acids, and small molecules. It leverages all experimentally available 3D

structures of query proteins from the PDB and their homologous proteins. Additionally, StructMAn can also utilize the models produced by AlphaFold. It calculates an interaction score that evaluates the potential impact of a mutation on protein structure and its interacting partners and is based on sequence identity, alignment length, and proximity to its interacting molecules. Further, StructMAn can also perform Gene Ontology (GO) term and pathway enrichment analyses on proteins containing mutations, providing insights into the biological processes, functions, and cellular components associated with these changes. This is useful in clinical genetics for differentiating benign and pathogenic mutations and identifying mutations linked to drug resistance.

MetaProFi, with its ability to efficiently index large-scale datasets along with its fast search features, offers significant advantages when integrated with StructMAn. By leveraging MetaProFi's capability to query NGS reads, contigs, or assembled genomes against an amino acid index, we can identify and analyze genetic variants from extensive metagenomics and microbiome datasets, for example. When nucleotide sequences are used as queries, MetaProFi identifies and extracts the open reading frames (ORFs), translates the ORFs to coding sequences, and then queries the amino acid index and extracts variants according to the Human Genome Variation Society (HGVS) nomenclature [321, 322], which provides a standardized format for describing DNA, RNA, and protein sequence variants.

The integration with StructMAn can be facilitated through the API already developed during the work presented in this thesis specifically for this purpose, allowing for seamless structural annotation and functional impact analysis of the identified variants. To further this project, steps have been taken by containerizing StructMAn within the d-StructMAn [323] project. This creates a scalable and efficient environment that can easily extend to include MetaProFi in the same container. Containerization ensures consistency across different computing environments, simplifies deployment, and enhances reproducibility. Through this integration, MetaProFi provides a powerful tool for indexing and querying genetic data and extends its functionality by leveraging StructMAn's capabilities to offer comprehensive insights into the structural and functional implications of genetic variants. This combined approach is valuable for distinguishing between benign and pathogenic mutations and

identifying mutations associated with drug resistance, thereby supporting advanced clinical genetics and personalized medicine research.

5.2.2 METAPROFI-BEACON INTEGRATION

As an extension, MetaProFi can be integrated with the Global Alliance for Genomics and Health (GA4GH) [324] Beacon project [325]. GA4GH is an international collaborative effort focused on creating standards for the responsible and secure sharing of genomic and clinical data that is essential for advancing personalized medicine. One of the key initiatives of GA4GH is the Beacon project, a federated system developed to enhance the findability, accessibility, and sharing of genomic data across different institutions while maintaining privacy. Through the Beacon project, we can query the federated and distributed genomic databases to determine the presence of specific genetic variants without centralizing the data. This approach ensures that the data remains with the original stewards, thereby promoting data security and privacy. Beacon's latest version [326] introduces advanced features, including metadata-rich queries and cohort analysis, which allows data to be filtered by various criteria, such as age, gender, and clinical annotations. Integrating MetaProFi with the Beacon project can significantly enhance personalized medicine's data discovery and analysis capabilities. We can develop an API allowing MetaProFi to seamlessly query Beacons for detailed information, including comprehensive clinical annotations and phenotypic data. MetaProFi can offer in-depth insights into the clinical and genetic implications of the variants identified by leveraging the features offered by the Beacon project.

5.2.3 METAPROFI AS A WEB SERVICE

All these tools and integrations can be developed into a web service and made publicly accessible. This service can offer prebuilt indexes of a large number of datasets, enabling users to query these indices efficiently through the web server. By using as a web-based platform, users can utilize MetaProFi's fast indexing and efficient sequence search capabilities, making exploring and analyzing large-scale genomic datasets easier. Integrating MetaProFi with StructMAn and the GA4GH Beacon

project allows for a comprehensive approach to genetic data analysis, encompassing variant identification, structural annotation, and clinical interpretation. This platform would support understanding the genetic basis of drug resistance, classifying variants as benign or pathogenic, and examining their structural and functional impacts. These functionalities can be further expanded by integrating them with the MD simulations methodology employed as a part of this thesis to study the effect of the identified variants on protein dynamics, structure, and conformational changes crucial for activation and ligand binding at the atomistic level. This platform would offer a complete pipeline, covering everything from sequence analysis and variant identification to structural annotation and classification of variants. It will also facilitate the extraction of clinical outcomes and the study of protein structural and conformational changes. Such a tool will be invaluable for providing detailed insights into the molecular mechanisms underlying genetic mutations, enhancing our understanding of their implications for personalized medicine and targeted therapies.

5.2.4 METAPROFI AS A GALAXY TOOL

MetaProFi can be integrated into the Galaxy platform [327] to offer significant benefits for analyzing large-scale sequence datasets. Galaxy is an open-source platform that enables scalable and reproducible data analysis across domains. Galaxy provides a robust, user-friendly interface for developing workflows and supports integration with numerous tools and datasets. Integrating MetaProFi involves creating a Bioconda [328] recipe and an XML wrapper for its command-line options, which can be submitted to Galaxy ToolShed [329]. This integration allows Galaxy servers worldwide to install MetaProFi, which could be an efficient alternative to resource-intensive tools like BLAST. Since public Galaxy servers provide free, shared computing resources, incorporating MetaProFi can significantly save computational resources and improve efficiency for hundreds of thousands of users. MetaProFi's unique indexing and querying capabilities, combined with Galaxy's workflow management and visualization features, would be a powerful tool for analyzing large-scale genomic datasets and performing various downstream analyses. This integration would greatly benefit researchers working with metage-

nomic and microbiome datasets and enable them to efficiently identify genetic variants and study their structural and functional implications (in combination with StructMAn).

BIBLIOGRAPHY

[1] C R Woese, O Kandler, and M L Wheelis. "Towards a natural system of organisms: proposal for the domains Archaea, Bacteria, and Eucarya." In: *Proceedings of the National Academy of Sciences of the United States of America* 87.12 (June 1990), pp. 4576–4579. (Visited on 07/20/2024).

[2] William Martin and Michael J Russell. "On the origins of cells: a hypothesis for the evolutionary transitions from abiotic geochemistry to chemoautotrophic prokaryotes, and from prokaryotes to nucleated cells." In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 358.1429 (Jan. 29, 2003), pp. 59–85. DOI: 10.1098/rstb.2002.1183. (Visited on 07/20/2024).

[3] Geoffrey M. Cooper. "The Origin and Evolution of Cells." In: Sinauer Associates, 2000. (Visited on 07/20/2024).

[4] Francis Crick. "Central Dogma of Molecular Biology." In: *Nature* 227.5258 (Aug. 1970), pp. 561–563. DOI: 10.1038/227561a0. (Visited on 07/20/2024).

[5] J. Craig Venter et al. "The Sequence of the Human Genome." In: *Science* 291.5507 (Feb. 16, 2001), pp. 1304–1351. DOI: 10.1126/science.1058040. (Visited on 08/23/2023).

[6] Rasko Leinonen et al. "The European Nucleotide Archive." In: *Nucleic Acids Research* 39 (Database issue Jan. 2011), pp. D28–D31. DOI: 10.1093/nar/gkq967. (Visited on 07/25/2021).

[7] Rasko Leinonen, Hideaki Sugawara, and Martin Shumway. "The Sequence Read Archive." In: *Nucleic Acids Research* 39 (Database issue Jan. 2011), pp. D19–D21. DOI: 10.1093/nar/gkq1019. (Visited on 07/25/2021).

[8] S. F. Altschul et al. "Basic local alignment search tool." In: *Journal of Molecular Biology* 215.3 (Oct. 5, 1990), pp. 403–410. DOI: 10.1016/S0022-2836(05)80360-2.

[9] Christian B. Anfinsen. "Principles that Govern the Folding of Protein Chains." In: *Science* 181.4096 (July 20, 1973), pp. 223–230. DOI: 10.1126/science.181.4096.223. (Visited on 07/25/2024).

[10] Sanjay K. Srikakulam, Tomas Bastys, and Olga V. Kalinina. "A shift of dynamic equilibrium between the KIT active and inactive states causes drug resistance." In: *Proteins: Structure, Function, and Bioinformatics* 88.11 (2020), pp. 1434–1446. DOI: 10.1002/prot.25963. (Visited on 03/07/2023).

[11] I De Beauchêne et al. "Hotspot Mutations in KIT Receptor Differentially Modulate Its Allosterically Coupled Conformational Dynamics: Impact on Activation and Drug Sensitivity." In: *PLoS Comput Biol* 10.7 (2014), p. 1003749. DOI: 10.1371/journal.pcbi.1003749.

[12] Maryam Abbaspour Babaei et al. "Receptor tyrosine kinase (c-Kit) inhibitors: a potential therapeutic target in cancer cells." In: *Drug Design, Development and Therapy* 10 (Aug. 1, 2016), pp. 2443–2459. DOI: 10.2147/DDDT.S89114. (Visited on 01/20/2018).

[13] Maria Debiec-Rychter et al. "Mechanisms of resistance to imatinib mesylate in gastrointestinal stromal tumors and activity of the PKC412 inhibitor against imatinib-resistant mutants." In: *Gastroenterology* 128.2 (Feb. 2005), pp. 270–279. DOI: 10.1053/J.GASTRO.2004.11.020.

[14] Georg Dultz et al. "Epistatic interactions promote persistence of NS3-Q80K in HCV infection by compensating for protein folding instability." In: *Journal of Biological Chemistry* 297.3 (Sept. 1, 2021), p. 101031. DOI: 10.1016/j.jbc.2021.101031. (Visited on 03/07/2023).

[15] Sanjay K Srikakulam et al. "MetaProFi: an ultrafast chunked Bloom filter for storing and querying protein and nucleotide sequence data for accurate identification of functionally relevant genetic variants." In: *Bioinformatics* 39.3 (Mar. 1, 2023), btad101. DOI: 10.1093/bioinformatics/btad101. (Visited on 07/29/2024).

[16] Stanley L. Miller. "A Production of Amino Acids Under Possible Primitive Earth Conditions." In: *Science* 117.3046 (May 15, 1953), pp. 528–529. DOI: 10.1126/science.117.3046.528. (Visited on 07/20/2024).

[17] Melanie Bengtson and Eric D. Edstrom. "A New Method for Testing Models of Prebiotic Peptide Assembly." In: *Advances in BioChirality*. Elsevier, 1999, pp. 115–123. ISBN: 978-0-08-043404-9. DOI: 10.1016/B978-008043404-9/50009-7. (Visited on 07/20/2024).

[18] Harold C. Urey. "On the Early Chemical History of the Earth and the Origin of Life." In: *Proceedings of the National Academy of Sciences of the United States of America* 38.4 (Apr. 1952), pp. 351–363. (Visited on 07/20/2024).

[19] Bruce Alberts et al. "The RNA World and the Origins of Life." In: Garland Science, 2002. (Visited on 07/20/2024).

[20] Jennifer A. Doudna and Thomas R. Cech. "The chemical repertoire of natural ribozymes." In: *Nature* 418.6894 (July 2002), pp. 222–228. DOI: 10.1038/418222a. (Visited on 07/20/2024).

[21] Gerald F. Joyce. "The antiquity of RNA-based evolution." In: *Nature* 418.6894 (July 2002), pp. 214–221. DOI: 10.1038/418214a. (Visited on 07/20/2024).

[22] Hyman Hartman and Alexei Fedorov. "The origin of the eukaryotic cell: A genomic investigation." In: *Proceedings of the National Academy of Sciences* 99.3 (Feb. 5, 2002), pp. 1420–1425. DOI: 10.1073/pnas.032658599. (Visited on 07/20/2024).

[23] T Vellai and G Vida. "The origin of eukaryotes: the difference between prokaryotic and eukaryotic cells." In: *Proceedings of the Royal Society B: Biological Sciences* 266.1428 (Aug. 7, 1999), pp. 1571–1577. (Visited on 07/20/2024).

[24] W. Ford Doolittle. "Phylogenetic Classification and the Universal Tree." In: *Science* 284.5423 (June 25, 1999), pp. 2124–2128. DOI: 10.1126/science.284.5423.2124. (Visited on 08/28/2024).

[25] Geoffrey M. Cooper. "The Origin and Evolution of Cells." In: Sinauer Associates, 2000. (Visited on 07/20/2024).

[26] Bruce Alberts et al. "The Diversity of Genomes and the Tree of Life." In: Garland Science, 2002. (Visited on 07/20/2024).

[27] Bruce Alberts et al. "How Cells Read the Genome: From DNA to Protein." In: Garland Science, 2002. (Visited on 07/20/2024).

[28] J. D. Watson and F. H. C. Crick. "Molecular Structure of Nucleic Acids: A Structure for Deoxyribose Nucleic Acid." In: *Nature* 171.4356 (Apr. 1953), pp. 737–738. DOI: 10.1038/171737a0. (Visited on 07/20/2024).

[29] Taishan Hu et al. "Next-generation sequencing technologies: An overview." In: *Human Immunology*. Next Generation Sequencing and its Application to Medical Laboratory Immunology 82.11 (Nov. 1, 2021), pp. 801–811. DOI: 10.1016/j.humimm.2021.02.012. (Visited on 08/23/2023).

[30] Thomas Mitchell-Olds, John H. Willis, and David B. Goldstein. "Which evolutionary processes influence natural genetic variation for phenotypic traits?" In: *Nature Reviews Genetics* 8.11 (Nov. 2007), pp. 845–856. DOI: 10.1038/nrg2207. (Visited on 08/11/2024).

[31] Lars Feuk, Andrew R. Carson, and Stephen W. Scherer. "Structural variation in the human genome." In: *Nature Reviews Genetics* 7.2 (Feb. 2006), pp. 85–97. DOI: 10.1038/nrg1767. (Visited on 07/23/2024).

[32] Ryan E. Mills et al. "An initial map of insertion and deletion (INDEL) variation in the human genome." In: *Genome Res.* 16.9 (Sept. 1, 2006), pp. 1182–1190. DOI: 10.1101/gr.4565806. (Visited on 08/28/2024).

[33] Donald F. Conrad et al. "Origins and functional impact of copy number variation in the human genome." In: *Nature* 464.7289 (Apr. 2010), pp. 704–712. DOI: 10.1038/nature08516. (Visited on 07/23/2024).

[34] Larisa Fedorova et al. "Analysis of Common SNPs across Continents Reveals Major Genomic Differences between Human Populations." In: *Genes* 13.8 (Aug. 18, 2022), p. 1472. DOI: 10.3390/genes13081472. (Visited on 07/23/2024).

[35] Adam Auton et al. "A global reference for human genetic variation." In: *Nature* 526.7571 (Oct. 2015), pp. 68–74. DOI: 10.1038/nature15393. (Visited on 08/28/2024).

[36] F. Sanger and E. O. P. Thompson. "The amino-acid sequence in the glycyl chain of insulin. 2. The investigation of peptides from

enzymic hydrolysates." In: *Biochemical Journal* 53.3 (Feb. 1, 1953), pp. 366–374. DOI: 10.1042/bj0530366. (Visited on 08/11/2024).

[37] Alice Maria Giani et al. "Long walk to genomics: History and current approaches to genome sequencing and assembly." In: *Computational and Structural Biotechnology Journal* 18 (Jan. 1, 2020), pp. 9–19. DOI: 10.1016/j.csbj.2019.11.002. (Visited on 07/21/2024).

[38] Robert W. Holley et al. "Structure of a Ribonucleic Acid." In: *Science* 147.3664 (Mar. 19, 1965), pp. 1462–1465. DOI: 10.1126/science.147.3664.1462. (Visited on 07/22/2024).

[39] F. Sanger, S. Nicklen, and A. R. Coulson. "DNA sequencing with chain-terminating inhibitors." In: *Proceedings of the National Academy of Sciences of the United States of America* 74.12 (Dec. 1977), pp. 5463–5467. DOI: 10.1073/pnas.74.12.5463.

[40] Simon T Bennett et al. "Toward the $1000 Human Genome." In: *Pharmacogenomics* 6.4 (July 1, 2005), pp. 373–382. DOI: 10.1517/14622416.6.4.373. (Visited on 08/28/2024).

[41] Lin Liu et al. "Comparison of Next-Generation Sequencing Systems." In: *BioMed Research International* 2012.1 (2012), p. 251364. DOI: 10.1155/2012/251364. (Visited on 07/21/2024).

[42] Jonathan M. Rothberg et al. "An integrated semiconductor device enabling non-optical genome sequencing." In: *Nature* 475.7356 (July 2011), pp. 348–352. DOI: 10.1038/nature10242. (Visited on 07/21/2024).

[43] Anthony Rhoads and Kin Fai Au. "PacBio Sequencing and Its Applications." In: *Genomics, Proteomics & Bioinformatics*. SI: Metagenomics of Marine Environments 13.5 (Oct. 1, 2015), pp. 278–289. DOI: 10.1016/j.gpb.2015.08.002. (Visited on 08/28/2024).

[44] John Eid et al. "Real-Time DNA Sequencing from Single Polymerase Molecules." In: *Science* 323.5910 (Jan. 2, 2009), pp. 133–138. DOI: 10.1126/science.1162986. (Visited on 07/21/2024).

[45] James Clarke et al. "Continuous base identification for single-molecule nanopore DNA sequencing." In: *Nature Nanotechnology* 4.4 (Apr. 2009), pp. 265–270. DOI: 10.1038/nnano.2009.12. (Visited on 07/21/2024).

[46] Andrew D. Johnson. "An extended IUPAC nomenclature code for polymorphic nucleic acids." In: *Bioinformatics* 26.10 (May 15, 2010), pp. 1386–1389. DOI: 10.1093/bioinformatics/btq098. (Visited on 07/21/2024).

[47] Heena Satam et al. "Next-Generation Sequencing Technology: Current Trends and Advancements." In: *Biology* 12.7 (July 13, 2023), p. 997. DOI: 10.3390/biology12070997. (Visited on 07/21/2024).

[48] A. M. Maxam and W. Gilbert. "A new method for sequencing DNA." In: *Proceedings of the National Academy of Sciences of the United States of America* 74.2 (Feb. 1977), pp. 560–564. DOI: 10.1073/pnas.74.2.560. (Visited on 08/23/2023).

[49] David A. Wheeler et al. "The complete genome of an individual by massively parallel DNA sequencing." In: *Nature* 452.7189 (Apr. 2008), pp. 872–876. DOI: 10.1038/nature06884. (Visited on 08/23/2023).

[50] Mehdi Kchouk, Jean Francois Gibrat, and Mourad Elloumi. "Generations of Sequencing Technologies: From First to Next Generation." In: *Biology and Medicine* 09.3 (2017). DOI: 10.4172/0974-8369.1000395. (Visited on 08/29/2024).

[51] Sara Goodwin, John D. McPherson, and W. Richard McCombie. "Coming of age: ten years of next-generation sequencing technologies." In: *Nature Reviews Genetics* 17.6 (June 2016), pp. 333–351. DOI: 10.1038/nrg.2016.49. (Visited on 08/23/2023).

[52] Fei Chen et al. "The History and Advances of Reversible Terminators Used in New Generations of Sequencing Technology." In: *Genomics, Proteomics & Bioinformatics* 11.1 (Feb. 1, 2013), pp. 34–40. DOI: 10.1016/j.gpb.2013.01.003. (Visited on 07/21/2024).

[53] Jonathan M. Rothberg and John H. Leamon. "The development and impact of 454 sequencing." In: *Nature Biotechnology* 26.10 (Oct. 2008), pp. 1117–1124. DOI: 10.1038/nbt1485. (Visited on 07/21/2024).

[54] Elena Espinosa et al. "Advancements in long-read genome sequencing technologies and algorithms." In: *Genomics* 116.3 (May 1, 2024), p. 110842. DOI: 10.1016/j.ygeno.2024.110842. (Visited on 07/21/2024).

[55] Richard J. Roberts, Mauricio O. Carneiro, and Michael C. Schatz. "The advantages of SMRT sequencing." In: *Genome Biology* 14.6 (July 3, 2013), p. 405. DOI: 10.1186/gb-2013-14-6-405. (Visited on 07/21/2024).

[56] Michael L. Metzker. "Sequencing technologies — the next generation." In: *Nature Reviews Genetics* 11.1 (Jan. 2010), pp. 31–46. DOI: 10.1038/nrg2626. (Visited on 08/23/2023).

[57] Todd J. Treangen and Steven L. Salzberg. "Repetitive DNA and next-generation sequencing: computational challenges and solutions." In: *Nature Reviews. Genetics* 13.1 (Nov. 29, 2011), pp. 36–46. DOI: 10.1038/nrg3117. (Visited on 07/22/2024).

[58] Can Alkan, Saba Sajjadian, and Evan E Eichler. "Limitations of next-generation genome sequence assembly." In: *Nature methods* 8.1 (Jan. 2011), pp. 61–65. DOI: 10.1038/nmeth.1527. (Visited on 07/22/2024).

[59] W R Pearson and D J Lipman. "Improved tools for biological sequence comparison." In: *Proceedings of the National Academy of Sciences of the United States of America* 85.8 (Apr. 1988), pp. 2444–2448. (Visited on 07/27/2024).

[60] Peter J. A. Cock et al. "The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants." In: *Nucleic Acids Research* 38.6 (Apr. 2010), pp. 1767–1771. DOI: 10.1093/nar/gkp1137. (Visited on 07/27/2024).

[61] Brent Ewing et al. "Base-Calling of Automated Sequencer Traces UsingPhred. I. Accuracy Assessment." In: (Mar. 1, 1998). DOI: 10.1101/gr.8.3.175. (Visited on 07/27/2024).

[62] Brent Ewing and Phil Green. "Base-Calling of Automated Sequencer Traces Using Phred. II. Error Probabilities." In: (Mar. 1, 1998). DOI: 10.1101/gr.8.3.186. (Visited on 07/27/2024).

[63] Susana Vinga and Jonas Almeida. "Alignment-free sequence comparison—a review." In: *Bioinformatics* 19.4 (Mar. 1, 2003), pp. 513–523. DOI: 10.1093/bioinformatics/btg005. (Visited on 08/23/2023).

[64] Paul Flicek and Ewan Birney. "Sense from sequence reads: methods for alignment and assembly." In: *Nature Methods* 6.11 (Nov. 2009), S6–S12. DOI: 10.1038/nmeth.1376. (Visited on 07/22/2024).

[65] Giulio Formenti et al. "The era of reference genomes in conservation genomics." In: (). DOI: 10.1016/j.tree.2021.11.008. (Visited on 07/23/2024).

[66] Riyue Bao et al. "Review of Current Methods, Applications, and Data Management for the Bioinformatics Analysis of Whole Exome Sequencing." In: *Cancer Informatics* 13s2 (Jan. 1, 2014), CIN.S13779. DOI: 10.4137/CIN.S13779. (Visited on 07/23/2024).

[67] Torsten Günther and Carl Nettelblad. "The presence and impact of reference bias on population genomic studies of prehistoric human populations." In: *PLOS Genetics* 15.7 (July 26, 2019), e1008302. DOI: 10.1371/journal.pgen.1008302. (Visited on 08/29/2024).

[68] Jeongkyu Kim, Mingeun Ji, and Gangman Yi. "A Review on Sequence Alignment Algorithms for Short Reads Based on Next-Generation Sequencing." In: *IEEE Access* 8 (2020), pp. 189811–189822. DOI: 10.1109/ACCESS.2020.3031159. (Visited on 07/22/2024).

[69] Heng Li and Nils Homer. "A survey of sequence alignment algorithms for next-generation sequencing." In: *Briefings in Bioinformatics* 11.5 (Sept. 1, 2010), pp. 473–483. DOI: 10.1093/bib/bbq015. (Visited on 07/22/2024).

[70] Matthew Ruffalo, Thomas LaFramboise, and Mehmet Koyutürk. "Comparative analysis of algorithms for next-generation sequencing read alignment." In: *Bioinformatics* 27.20 (Oct. 15, 2011), pp. 2790–2796. DOI: 10.1093/bioinformatics/btr477. (Visited on 07/22/2024).

[71] Qanita Bani Baker et al. "Comprehensive comparison of cloud-based NGS data analysis and alignment tools." In: *Informatics in Medicine Unlocked* 18 (Jan. 1, 2020), p. 100296. DOI: 10.1016/j.imu.2020.100296. (Visited on 07/22/2024).

[72] Saul B. Needleman and Christian D. Wunsch. "A general method applicable to the search for similarities in the amino acid sequence of two proteins." In: *Journal of Molecular Biology* 48.3 (Mar. 28, 1970), pp. 443–453. DOI: 10.1016/0022-2836(70)90057-4. (Visited on 04/04/2022).

[73] T. F. Smith and M. S. Waterman. "Identification of common molecular subsequences." In: *Journal of Molecular Biology* 147.1 (Mar. 25, 1981), pp. 195–197. DOI: 10.1016/0022-2836(81)90087-5. (Visited on 08/29/2024).

[74] S Henikoff and J G Henikoff. "Amino acid substitution matrices from protein blocks." In: *Proceedings of the National Academy of Sciences of the United States of America* 89.22 (Nov. 15, 1992), pp. 10915–10919. (Visited on 07/23/2024).

[75] M. O. Dayhoff and R. M. Schwartz. "Chapter 22: A model of evolutionary change in proteins." In: *in Atlas of Protein Sequence and Structure*. 1978.

[76] Heng Li and Richard Durbin. "Fast and accurate short read alignment with Burrows–Wheeler transform." In: *Bioinformatics* 25.14 (July 15, 2009), pp. 1754–1760. DOI: 10.1093/bioinformatics/btp324. (Visited on 07/22/2024).

[77] Ben Langmead et al. "Ultrafast and memory-efficient alignment of short DNA sequences to the human genome." In: *Genome Biology* 10.3 (Mar. 4, 2009), R25. DOI: 10.1186/gb-2009-10-3-r25. (Visited on 07/22/2024).

[78] M. Burrows et al. "A block-sorting lossless data compression algorithm." In: *SRS Research Report* 124 (1994). (Visited on 08/29/2024).

[79] P. Ferragina and G. Manzini. "Opportunistic data structures with applications." In: *Proceedings 41st Annual Symposium on Foundations of Computer Science*. 41st Annual Symposium on Foundations of Computer Science. Redondo Beach, CA, USA: IEEE Comput. Soc, 2000, pp. 390–398. ISBN: 978-0-7695-0850-4. DOI: 10.1109/SFCS.2000.892127. (Visited on 07/23/2024).

[80] Ben Langmead and Steven L. Salzberg. "Fast gapped-read alignment with Bowtie 2." In: *Nature Methods* 9.4 (Apr. 2012), pp. 357–359. DOI: 10.1038/nmeth.1923. (Visited on 07/23/2024).

[81] Sara El-Metwally et al. "Next-Generation Sequence Assembly: Four Stages of Data Processing and Computational Challenges." In: *PLOS Computational Biology* 9.12 (Dec. 12, 2013), e1003345. DOI: 10.1371/journal.pcbi.1003345. (Visited on 08/29/2024).

[82] Andrzej Zielezinski et al. "Alignment-free sequence comparison: benefits, applications, and tools." In: *Genome Biology* 18.1 (Oct. 3, 2017), p. 186. DOI: 10.1186/s13059-017-1319-7. (Visited on 07/23/2024).

[83] Kai Song et al. "New developments of alignment-free sequence comparison: measures, statistics and next-generation sequencing." In: *Briefings in Bioinformatics* 15.3 (May 1, 2014), pp. 343–353. DOI: 10.1093/bib/bbt067. (Visited on 07/23/2024).

[84] Andrzej Zielezinski et al. "Benchmarking of alignment-free sequence comparison methods." In: *Genome Biology* 20.1 (July 25, 2019), pp. 1–18. DOI: 10.1186/s13059-019-1755-7. (Visited on 07/23/2024).

[85] Yiming He et al. "De novo assembly methods for next generation sequencing data." In: *Tsinghua Science and Technology* 18.5 (Oct. 2013), pp. 500–514. DOI: 10.1109/TST.2013.6616523. (Visited on 07/23/2024).

[86] David Hernandez et al. "De novo bacterial genome sequencing: Millions of very short reads assembled on a desktop computer." In: (May 1, 2008). DOI: 10.1101/gr.072033.107. (Visited on 07/23/2024).

[87] Melissa de la Bastide and W. Richard McCombie. "Assembling Genomic DNA Sequences with PHRAP." In: *Current Protocols in Bioinformatics* 17.1 (2007), pp. 11.4.1–11.4.15. DOI: 10.1002/0471250953.bi1104s17. (Visited on 07/23/2024).

[88] Xiaoqiu Huang and Anup Madan. "CAP3: A DNA Sequence Assembly Program." In: *Genome Research* 9.9 (Sept. 1999), pp. 868–877. (Visited on 07/23/2024).

[89] Eugene W. Myers et al. "A Whole-Genome Assembly of Drosophila." In: *Science* 287.5461 (Mar. 24, 2000), pp. 2196–2204. DOI: 10.1126/science.287.5461.2196. (Visited on 08/29/2024).

[90] Pavel A. Pevzner, Haixu Tang, and Michael S. Waterman. "An Eulerian path approach to DNA fragment assembly." In: *Proceedings of the National Academy of Sciences of the United States of America* 98.17 (Aug. 14, 2001), pp. 9748–9753. DOI: 10.1073/pnas.171285098. (Visited on 07/23/2024).

[91] Daniel R. Zerbino and Ewan Birney. "Velvet: Algorithms for de novo short read assembly using de Bruijn graphs." In: *Genome Research* 18.5 (May 2008), pp. 821–829. DOI: 10.1101/gr.074492.107. (Visited on 07/23/2024).

[92] Jared T. Simpson et al. "ABySS: A parallel assembler for short read sequence data." In: *Genome Research* 19.6 (June 2009), pp. 1117–1123. DOI: 10.1101/gr.089532.108. (Visited on 07/23/2024).

[93] Anton Bankevich et al. "SPAdes: A New Genome Assembly Algorithm and Its Applications to Single-Cell Sequencing." In: *Journal of Computational Biology* 19.5 (May 2012), pp. 455–477. DOI: 10.1089/cmb.2012.0021. (Visited on 07/23/2024).

[94] Kai Song, Jie Ren, and Fengzhu Sun. "Reads Binning Improves Alignment-Free Metagenome Comparison." In: *Frontiers in Genetics* 10 (Nov. 21, 2019), p. 1156. DOI: 10.3389/fgene.2019.01156. (Visited on 08/29/2024).

[95] Jie Ren et al. "Alignment-Free Sequence Analysis and Applications." In: (July 20, 2018). DOI: 10.1146/annurev-biodatasci-080917-013431. (Visited on 07/23/2024).

[96] Guillaume Marçais and Carl Kingsford. "A fast, lock-free approach for efficient parallel counting of occurrences of k-mers." In: *Bioinformatics* 27.6 (Mar. 15, 2011), pp. 764–770. DOI: 10.1093/bioinformatics/btr011. (Visited on 07/23/2024).

[97] Guillaume Rizk, Dominique Lavenier, and Rayan Chikhi. "DSK: k-mer counting with very low memory usage." In: *Bioinformatics* 29.5 (Mar. 1, 2013), pp. 652–653. DOI: 10.1093/bioinformatics/btt020. (Visited on 07/23/2024).

[98] Sebastian Deorowicz et al. "KMC 2: fast and resource-frugal k-mer counting." In: *Bioinformatics* 31.10 (May 15, 2015), pp. 1569–1576. DOI: 10.1093/bioinformatics/btv022. (Visited on 07/23/2024).

[99] Nathan A Ahlgren et al. "Alignment-free d_2^* oligonucleotide frequency dissimilarity measure improves prediction of hosts from metagenomically-derived viral sequences." In: *Nucleic Acids Research* 45.1 (Jan. 9, 2017), pp. 39–53. DOI: 10.1093/nar/gkw1002. (Visited on 08/29/2024).

[100] Ji Qi, Hong Luo, and Bailin Hao. "CVTree: a phylogenetic tree reconstruction tool based on whole genomes." In: *Nucleic Acids Research* 32 (Web Server issue July 1, 2004), W45–W47. DOI: 10.1093/nar/gkh362. (Visited on 07/23/2024).

[101] Gesine Reinert et al. "Alignment-Free Sequence Comparison (I): Statistics and Power." In: *Journal of Computational Biology* 16.12 (Dec. 2009), pp. 1615–1634. DOI: 10.1089/cmb.2009.0198. (Visited on 07/23/2024).

[102] Lin Wan et al. "Alignment-Free Sequence Comparison (II): Theoretical Power of Comparison Statistics." In: *Journal of Computational Biology* 17.11 (Nov. 2010), pp. 1467–1490. DOI: 10.1089/cmb.2010.0056. (Visited on 07/23/2024).

[103] Gregory E. Sims et al. "Alignment-free genome comparison with feature frequency profiles (FFP) and optimal resolutions." In: *Proceedings of the National Academy of Sciences of the United States of America* 106.8 (Feb. 24, 2009), pp. 2677–2682. DOI: 10.1073/pnas.0813249106. (Visited on 07/23/2024).

[104] JaeJin Choi and Sung-Hou Kim. "A genome Tree of Life for the Fungi kingdom." In: *Proceedings of the National Academy of Sciences of the United States of America* 114.35 (Aug. 29, 2017), pp. 9391–9396. DOI: 10.1073/pnas.1711939114. (Visited on 07/23/2024).

[105] Bruce Alberts et al. *Molecular Biology of the Cell*. 4th. Garland Science, 2002. ISBN: 978-0-8153-3218-3 978-0-8153-4072-0.

[106] G. N. Ramachandran and V. Sasisekharan. "Conformation of Polypeptides and Proteins*†." In: *Advances in Protein Chemistry*. Ed. by C. B. Anfinsen et al. Vol. 23. Academic Press, Jan. 1, 1968, pp. 283–437. DOI: 10.1016/S0065-3233(08)60402-7. (Visited on 07/24/2024).

[107] T E Creighton. "Protein folding." In: *Biochemical Journal* 270.1 (Aug. 15, 1990), pp. 1–16. DOI: 10.1042/bj2700001. (Visited on 08/29/2024).

[108] M Beissinger and J Buchner. "How chaperones fold proteins." In: *Biological chemistry* 379.3 (Mar. 1, 1998), pp. 245–259. (Visited on 04/16/2023).

[109] Peter D. Sun, Christine E. Foster, and Jeffrey C. Boyington. "Overview of Protein Structural and Functional Folds." In: *Current Protocols in Protein Science* 35.1 (Feb. 2004), pp. 1711–171189. DOI: 10.1002/0471140864.ps1701s35. (Visited on 07/25/2024).

[110] Wright Pe and Dyson Hj. "Intrinsically unstructured proteins: re-assessing the protein structure-function paradigm." In: *PubMed* (1999). (Visited on 07/26/2024).

[111] Linus Pauling, Robert B. Corey, and H. R. Branson. "The structure of proteins: Two hydrogen-bonded helical configurations of the polypeptide chain." In: *Proceedings of the National Academy of Sciences* 37.4 (Apr. 1951), pp. 205–211. DOI: 10.1073/pnas.37.4.205. (Visited on 07/26/2024).

[112] Linus Carl Pauling, R. B. Corey, and William Thomas Astbury. "Stable configurations of polypeptide chains." In: *Proceedings of the Royal Society of London. Series B - Biological Sciences* 141.902 (Jan. 1997), pp. 21–33. DOI: 10.1098/rspb.1953.0012. (Visited on 07/26/2024).

[113] Philip E. Bourne and Helge Weissig, eds. *Structural bioinformatics*. Methods of biochemical analysis v. 44. Hoboken, N.J: Wiley-Liss, 2003. 649 pp. ISBN: 978-0-471-20200-4 978-0-471-20199-1. (Visited on 07/08/2023).

[114] Janet M Thornton et al. "Protein folds, functions and evolution." In: *Journal of Molecular Biology* 293.2 (Oct. 22, 1999), pp. 333–342. DOI: 10.1006/jmbi.1999.3054. (Visited on 08/12/2024).

[115] Andrew Cr Martin et al. "Protein folds and functions." In: *Structure* 6.7 (July 1998), pp. 875–884. DOI: 10.1016/S0969-2126(98)00089-6. (Visited on 08/29/2024).

[116] Milo M. Lin and Ahmed H. Zewail. "Hydrophobic forces and the length limit of foldable protein domains." In: *Proceedings of the National Academy of Sciences* 109.25 (June 19, 2012), pp. 9851–9856. DOI: 10.1073/pnas.1207382109. (Visited on 07/26/2024).

[117] T J Hubbard et al. "SCOP: a Structural Classification of Proteins database." In: *Nucleic Acids Research* 27.1 (Jan. 1, 1999), pp. 254–256. (Visited on 07/26/2024).

[118] R. Dustin Schaeffer and Valerie Daggett. "Protein folds and protein folding." In: *Protein Engineering, Design and Selection* 24.1 (Jan. 1, 2011), pp. 11–19. DOI: 10.1093/protein/gzq096. (Visited on 08/12/2024).

[119] Konstantinos Sousounis et al. "Conservation of the three-dimensional structure in non-homologous or unrelated proteins." In: *Human Genomics* 6.1 (Aug. 2, 2012), p. 10. DOI: 10.1186/1479-7364-6-10. (Visited on 07/26/2024).

[120] Walter R. P. Novak. "Tertiary Structure Domains, Folds, and Motifs." In: *Molecular Life Sciences*. Ed. by Ellis Bell. New York, NY: Springer New York, 2014, pp. 1–5. ISBN: 978-1-4614-6436-5. DOI: 10.1007/978-1-4614-6436-5_15-3. (Visited on 07/26/2024).

[121] H. Sund and K. Weber. "The Quaternary Structure of Proteins." In: *Angewandte Chemie International Edition in English* 5.2 (1966), pp. 231–245. DOI: 10.1002/anie.196602311. (Visited on 07/26/2024).

[122] Gregory R Bowman, Vincent A Voelz, and Vijay S Pande. "Taming the complexity of protein folding." In: *Current Opinion in Structural Biology* 21.1 (Feb. 1, 2011), pp. 4–11. DOI: 10.1016/j.sbi.2010.10.006. (Visited on 07/26/2024).

[123] J. C. Kendrew et al. "A Three-Dimensional Model of the Myoglobin Molecule Obtained by X-Ray Analysis." In: *Nature* 181.4610 (Mar. 1958), pp. 662–666. DOI: 10.1038/181662a0. (Visited on 07/26/2024).

[124] Bruce Alberts et al. "The Shape and Structure of Proteins." In: Garland Science, 2002. (Visited on 07/24/2024).

[125] M S Smyth and J H J Martin. "x Ray crystallography." In: *Molecular Pathology* 53.1 (Feb. 2000), pp. 8–14. (Visited on 07/26/2024).

[126] K Wüthrich. "Protein structure determination in solution by NMR spectroscopy." In: *Journal of Biological Chemistry* 265.36 (Dec. 25, 1990), pp. 22059–22062. DOI: 10.1016/S0021-9258(18)45665-7. (Visited on 07/26/2024).

[127] Ka Man Yip et al. "Atomic-resolution protein structure determination by cryo-EM." In: *Nature* 587.7832 (Nov. 2020), pp. 157–161. DOI: 10.1038/s41586-020-2833-4. (Visited on 07/26/2024).

[128] Marta Carroni and Helen R. Saibil. "Cryo electron microscopy to determine the structure of macromolecular complexes." In: *Methods (San Diego, Calif.)* 95 (Feb. 15, 2016), pp. 78–85. DOI: 10.1016/j.ymeth.2015.11.023. (Visited on 07/26/2024).

[129] Bissan Al-Lazikani et al. "Protein structure prediction." In: *Current Opinion in Chemical Biology* 5.1 (Feb. 1, 2001), pp. 51–56. DOI: 10.1016/S1367-5931(00)00164-2. (Visited on 07/27/2024).

[130] Yang Zhang. "Protein structure prediction: when is it useful?" In: *Current Opinion in Structural Biology*. Theory and simulation / Macromolecular assemblages 19.2 (Apr. 1, 2009), pp. 145–155. DOI: 10.1016/j.sbi.2009.02.005. (Visited on 07/27/2024).

[131] John Jumper et al. "Highly accurate protein structure prediction with αFold." In: *Nature* 596.7873 (Aug. 2021), pp. 583–589. DOI: 10.1038/s41586-021-03819-2. (Visited on 07/28/2024).

[132] Andrej Šali and Tom L. Blundell. "Comparative Protein Modelling by Satisfaction of Spatial Restraints." In: *Journal of Molecular Biology* 234.3 (Dec. 5, 1993), pp. 779–815. DOI: 10.1006/jmbi.1993.1626. (Visited on 07/27/2024).

[133] Szymon Kaczanowski and Piotr Zielenkiewicz. "Why similar protein sequences encode similar three-dimensional structures?" In: *Theoretical Chemistry Accounts* 125.3 (Mar. 2010), pp. 643–650. DOI: 10.1007/s00214-009-0656-3. (Visited on 07/27/2024).

[134] C Chothia and A M Lesk. "The relation between the divergence of sequence and structure in proteins." In: *The EMBO Journal* 5.4 (Apr. 1986), pp. 823–826. (Visited on 07/27/2024).

[135] Helen M. Berman et al. "The Protein Data Bank." In: *Nucleic Acids Research* 28.1 (Jan. 1, 2000), pp. 235–242. DOI: 10.1093/nar/28.1.235. (Visited on 07/27/2024).

[136] Andrew Waterhouse et al. "SWISS-MODEL: homology modelling of protein structures and complexes." In: *Nucleic Acids Research* 46 (W1 July 2, 2018), W296–W303. DOI: 10.1093/nar/gky427. (Visited on 07/27/2024).

[137] Sali A and Blundell Tl. "Comparative protein modelling by satisfaction of spatial restraints." In: *PubMed* (1993). (Visited on 07/27/2024).

[138] Roberto Sánchez and Andrej Šali. "Comparative Protein Structure Modeling: Introduction and Practical Examples with Modeller." In: *Protein Structure Prediction: Methods and Protocols*. Ed. by David M. Webster. Totowa, NJ: Humana Press, 2000, pp. 97–129. ISBN: 978-1-59259-368-2. DOI: 10.1385/1-59259-368-2:97. (Visited on 05/09/2023).

[139] Corey Hardin, Taras V Pogorelov, and Zaida Luthey-Schulten. "*Ab initio* protein structure prediction." In: *Current Opinion in Structural Biology* 12.2 (Apr. 1, 2002), pp. 176–181. DOI: 10.1016/S0959-440X(02)00306-8. (Visited on 07/27/2024).

[140] David J Osguthorpe. "*Ab initio* protein folding." In: *Current Opinion in Structural Biology* 10.2 (Apr. 1, 2000), pp. 146–152. DOI: 10.1016/S0959-440X(00)00067-1. (Visited on 07/27/2024).

[141] M. E. Clamp et al. "Hybrid Monte Carlo: An efficient algorithm for condensed matter simulation." In: *Journal of Computational Chemistry* 15.8 (1994), pp. 838–846. DOI: 10.1002/jcc.540150805. (Visited on 07/27/2024).

[142] Jianyi Yang et al. "The I-TASSER Suite: protein structure and function prediction." In: *Nature Methods* 12.1 (Jan. 2015), pp. 7–8. DOI: 10.1038/nmeth.3213. (Visited on 07/27/2024).

[143] Sergey Ovchinnikov et al. "Large-scale determination of previously unsolved protein structures using evolutionary information." In: *eLife* 4 (Sept. 3, 2015). Ed. by Yibing Shan, e09248. DOI: 10.7554/eLife.09248. (Visited on 07/27/2024).

[144] J. Andrew McCammon, Bruce R. Gelin, and Martin Karplus. "Dynamics of folded proteins." In: *Nature* 267.5612 (June 1977), pp. 585–590. DOI: 10.1038/267585a0. (Visited on 07/27/2024).

[145] Martin Karplus and J. Andrew McCammon. "Molecular dynamics simulations of biomolecules." In: *Nature Structural Biology* 9.9 (Sept. 2002), pp. 646–652. DOI: 10.1038/nsb0902-646. (Visited on 07/27/2024).

[146] Scott A. Hollingsworth and Ron O. Dror. "Molecular dynamics simulation for all." In: *Neuron* 99.6 (Sept. 19, 2018), pp. 1129–1143. DOI: 10.1016/j.neuron.2018.08.011. (Visited on 07/27/2024).

[147] Stewart N Loh. "The missing Zinc: p53 misfolding and cancer." In: *Metallomics* 2.7 (July 1, 2010), pp. 442–449. DOI: 10.1039/c003915b. (Visited on 07/28/2024).

[148] Miao Wang and Randal J. Kaufman. "The impact of the endoplasmic reticulum protein-folding environment on cancer development." In: *Nature Reviews Cancer* 14.9 (Sept. 2014), pp. 581–597. DOI: 10.1038/nrc3800. (Visited on 07/28/2024).

[149] The UniProt Consortium. "UniProt: the Universal Protein Knowledgebase in 2023." In: *Nucleic Acids Research* 51 (D1 Jan. 6, 2023), pp. D523–D531. DOI: 10.1093/nar/gkac1052. (Visited on 07/27/2024).

[150] Vivien Junker et al. "The role SWISS-PROT and TrEMBL play in the genome research environment." In: *Journal of Biotechnology*. Genome Research Meets Biotechnology 78.3 (Mar. 31, 2000), pp. 221–234. DOI: 10.1016/S0168-1656(00)00198-X. (Visited on 07/27/2024).

[151] Viktor Hornak et al. "Comparison of multiple Amber force fields and development of improved protein backbone parameters." In: *Proteins: Structure, Function, and Bioinformatics* 65.3 (2006), pp. 712–725. DOI: 10.1002/prot.21123. (Visited on 08/30/2024).

[152] Fernando Martín-García et al. "Comparing Molecular Dynamics Force Fields in the Essential Subspace." In: *PLOS ONE* 10.3 (Mar. 2015). Ed. by Danilo Roccatano, e0121114. DOI: 10.1371/journal.pone.0121114. (Visited on 10/21/2023).

[153] Jing Huang et al. "CHARMM36m: An Improved Force Field for Folded and Intrinsically Disordered Proteins." In: *Nature methods* 14.1 (Jan. 2017), pp. 71–73. DOI: 10.1038/nmeth.4067. (Visited on 09/18/2019).

[154] Shruti Agarwal, Julhash U. Kazi, and Lars Rönnstrand. "Phosphorylation of the Activation Loop Tyrosine 823 in c-Kit Is Crucial for Cell Survival and Proliferation *." In: *Journal of Biological Chemistry* 288.31 (Aug. 2, 2013), pp. 22460–22468. DOI: 10.1074/jbc.M113.474072. (Visited on 08/30/2024).

[155] Ann D Kwong et al. "Hepatitis C virus NS3/4A protease." In: *Antiviral Research* 40.1 (Dec. 1, 1998), pp. 1–18. DOI: 10.1016/S0166-3542(98)00043-6. (Visited on 08/22/2023).

[156] Clifford D Mol et al. "Structural basis for the autoinhibition and STI-571 inhibition of c-Kit tyrosine kinase." In: *The Journal of biological chemistry* 279.30 (July 2004), pp. 31655–63. DOI: 10.1074/jbc.M403319200. (Visited on 09/21/2023).

[157] Clifford D Mol et al. "Structure of a c-kit product complex reveals the basis for kinase transactivation." In: *The Journal of biological chemistry* 278.34 (Aug. 2003), pp. 31461–4. DOI: 10.1074/jbc.C300186200. (Visited on 09/21/2023).

[158] Narayanan Eswar et al. "Protein Structure Modeling with MODELLER." In: Humana Press, 2008, pp. 145–159. DOI: 10.1007/978-1-60327-058-8_8. (Visited on 09/21/2023).

[159] Marc A. Martí-Renom et al. "Comparative Protein Structure Modeling of Genes and Genomes." In: *Annual Review of Biophysics* 29 (Volume 29, 2000 June 1, 2000), pp. 291–325. DOI: 10.1146/annurev.biophys.29.1.291. (Visited on 09/13/2024).

[160] Sunhwan Jo et al. "CHARMM-GUI: A web-based graphical user interface for CHARMM." In: *Journal of Computational Chemistry* 29.11 (2008), pp. 1859–1865. DOI: 10.1002/jcc.20945. (Visited on 04/08/2020).

[161] Sunhwan Jo et al. "CHARMM-GUI PDB Manipulator for Advanced Modeling and Simulations of Proteins Containing Nonstandard Residues." In: *Advances in protein chemistry and structural biology* 96 (2014), pp. 235–265. DOI: 10.1016/bs.apcsb.2014.06.002. (Visited on 04/08/2020).

[162] Jumin Lee et al. "CHARMM-GUI Input Generator for NAMD, GROMACS, AMBER, OpenMM, and CHARMM/OpenMM Simulations Using the CHARMM36 Additive Force Field." In: *Journal of Chemical Theory and Computation* 12.1 (Jan. 12, 2016), pp. 405–413. DOI: 10.1021/acs.jctc.5b00935. (Visited on 04/08/2020).

[163] Joost Schymkowitz et al. "The FoldX web server: an online force field." In: *Nucleic Acids Research* 33 (Web Server issue July 1, 2005), W382–388. DOI: 10.1093/nar/gki387. (Visited on 09/08/2023).

[164] N. Yao et al. "Molecular views of viral polyprotein processing revealed by the crystal structure of the hepatitis C virus bifunctional protease-helicase." In: *Structure (London, England: 1993)* 7.11

(Nov. 15, 1999), pp. 1353–1363. DOI: 10.1016/s0969-2126(00)80025-8. (Visited on 09/10/2023).

[165] David Van Der Spoel et al. "GROMACS: Fast, Flexible, and Free." In: *J Comput Chem* 26 (2005), pp. 1701–1718. DOI: 10.1002/jcc.20291. (Visited on 06/10/2019).

[166] In Suk Joung and Thomas E. III Cheatham. "Determination of Alkali and Halide Monovalent Ion Parameters for Use in Explicitly Solvated Biomolecular Simulations." In: *The Journal of Physical Chemistry B* 112.30 (July 1, 2008), pp. 9020–9041. DOI: 10.1021/jp8001614. (Visited on 08/30/2024).

[167] Giovanni Bussi, Davide Donadio, and Michele Parrinello. "Canonical sampling through velocity rescaling." In: *The Journal of Chemical Physics* 126.1 (Jan. 2007), p. 014101. DOI: 10.1063/1.2408420. (Visited on 08/05/2017).

[168] M. Parrinello and A. Rahman. "Polymorphic transitions in single crystals: A new molecular dynamics method." In: *Journal of Applied Physics* 52.12 (Dec. 1981), pp. 7182–7190. DOI: 10.1063/1.328693. (Visited on 08/05/2017).

[169] W F Van Gunsteren and H J C Berendsen. "Molecular Simulation A Leap-frog Algorithm for Stochastic Dynamics A LEAP-FROG ALGORITHM FOR STOCHASTIC DYNAMICS." In: *Molecular Simulation* 1 (1988), pp. 173–185. DOI: 10.1080/08927028808080941. (Visited on 08/05/2017).

[170] Tom Darden, Darrin York, and Lee Pedersen. "Particle mesh Ewald: An N · ln(N) method for Ewald sums in large systems." In: *The Journal of Chemical Physics* 98.12 (June 1993), pp. 10089–10092. DOI: 10.1063/1.464397. (Visited on 08/05/2017).

[171] Berk Hess. "P-LINCS: A Parallel Linear Constraint Solver for Molecular Simulation." In: (2008). DOI: 10.1021/ct700200b. (Visited on 08/05/2017).

[172] Wolfgang Kabsch and Christian Sander. "Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features." In: *Biopolymers* 22.12 (Dec. 1983), pp. 2577–2637. DOI: 10.1002/bip.360221211. (Visited on 08/07/2017).

[173] Patrick Kunzmann and Kay Hamacher. "Biotite: a unifying open source computational biology framework in Python." In: *BMC Bioinformatics* 19.1 (Oct. 1, 2018), p. 346. DOI: 10.1186/s12859-018-2367-z. (Visited on 04/07/2020).

[174] R Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing, 2017. (Visited on 03/05/2024).

[175] Nadezhda T. Doncheva et al. "Analyzing and visualizing residue networks of protein structures." In: *Trends in Biochemical Sciences* 36.4 (Apr. 2011), pp. 179–182. DOI: 10.1016/j.tibs.2011.01.002. (Visited on 04/07/2024).

[176] Vytautas Gapsys and Bert L. de Groot. "Optimal Superpositioning of Flexible Molecule Ensembles." In: *Biophysical Journal* 104.1 (Jan. 2013), pp. 196–207. DOI: 10.1016/j.bpj.2012.11.003. (Visited on 08/05/2020).

[177] Christopher L. McClendon et al. "Quantifying Correlations Between Allosteric Sites in Thermodynamic Ensembles." In: *Journal of Chemical Theory and Computation* 5.9 (Sept. 2009), pp. 2486–2502. DOI: 10.1021/ct9001812. (Visited on 09/21/2023).

[178] Tomas Bastys et al. "Consistent Prediction of Mutation Effect on Drug Binding in HIV-1 Protease Using Alchemical Calculations." In: *Journal of Chemical Theory and Computation* 14.7 (July 10, 2018), pp. 3397–3408. DOI: 10.1021/acs.jctc.7b01109. (Visited on 04/05/2020).

[179] Tatyana Krivobokova et al. "Partial Least-Squares Functional Mode Analysis: Application to the Membrane Proteins AQP1, Aqy1, and CLC-ec1." In: *Biophysj* 103 (2012), pp. 786–796. DOI: 10.1016/j.bpj.2012.07.022. (Visited on 03/21/2019).

[180] Robert Tibshirani, Guenther Walther, and Trevor Hastie. "Estimating the Number of Clusters in a Data Set Via the Gap Statistic." In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 63.2 (July 1, 2001), pp. 411–423. DOI: 10.1111/1467-9868.00293. (Visited on 08/30/2024).

[181] David J. Ketchen Jr. and Christopher L. Shook. "The application of cluster analysis in strategic management research: an analysis and critique." In: *Strategic Management Journal* 17.6 (June 1996), pp. 441–458. DOI: 10.1002/(SICI)1097-0266(199606)17:6<441::AID-SMJ819>3.0.CO;2-G. (Visited on 09/21/2023).

[182] Stevan R. Hubbard and W. Todd Miller. "Receptor tyrosine kinases: mechanisms of activation and signaling." In: *Current opinion in cell biology* 19.2 (Apr. 2007), pp. 117–123. DOI: 10.1016/j.ceb.2007.02.010. (Visited on 01/20/2018).

[183] Yi-fan Chen and Li-wu Fu. "Mechanisms of acquired resistance to tyrosine kinase inhibitors." In: *Acta Pharmaceutica Sinica B* 1.4 (Dec. 2011), pp. 197–207. DOI: 10.1016/j.apsb.2011.10.007. (Visited on 09/05/2017).

[184] Manoj Kumar Kashyap and Omar Abdel-Rahman. "Expression, regulation and targeting of receptor tyrosine kinases in esophageal squamous cell carcinoma." In: *Molecular Cancer* 17.1 (Feb. 19, 2018), pp. 1–11. DOI: 10.1186/s12943-018-0790-4. (Visited on 08/06/2024).

[185] Kalpana K. Bhanumathy et al. "Protein Tyrosine Kinases: Their Roles and Their Targeting in Leukemia." In: *Cancers* 13.2 (Jan. 7, 2021), p. 184. DOI: 10.3390/cancers13020184. (Visited on 08/06/2024).

[186] Kodappully S. Siveen et al. "Role of Non Receptor Tyrosine Kinases in Hematological Malignances and its Targeting by Natural Products." In: *Molecular Cancer* 17.1 (Feb. 19, 2018), pp. 1–21. DOI: 10.1186/s12943-018-0788-y. (Visited on 08/06/2024).

[187] Mark A Lemmon and Joseph Schlessinger. "Cell signaling by receptor-tyrosine kinases." In: *Cell* 141.7 (2010), pp. 1117–1134. DOI: 10.1016/j.cell.2010.06.011. (Visited on 10/24/2023).

[188] Tarik Regad. "Targeting RTK Signaling Pathways in Cancer." In: *Cancers* 7.3 (Sept. 3, 2015), pp. 1758–1784. DOI: 10.3390/cancers7030860. (Visited on 08/09/2024).

[189] Andreas Gschwind, Oliver M. Fischer, and Axel Ullrich. "The discovery of receptor tyrosine kinases: targets for cancer therapy." In: *Nature Reviews Cancer* 4.5 (May 2004), pp. 361–370. DOI: 10.1038/nrc1360. (Visited on 08/09/2024).

[190] Frédéric G. Brunet, Jean-Nicolas Volff, and Manfred Schartl. "Whole Genome Duplications Shaped the Receptor Tyrosine Kinase Repertoire of Jawed Vertebrates." In: *Genome Biology and Evolution* 8.5 (May 1, 2016), pp. 1600–1613. DOI: 10.1093/gbe/evw103. (Visited on 08/09/2024).

[191] Deborah L. Cadena and Gordon N. Gill. "Receptor tyrosine kinases." In: *The FASEB Journal* 6.6 (1992), pp. 2332–2337. DOI: 10.1096/fasebj.6.6.1312047. (Visited on 08/09/2024).

[192] Zhenfang Du and Christine M. Lovly. "Mechanisms of receptor tyrosine kinase activation in cancer." In: *Molecular Cancer* 17.1 (Feb. 19, 2018), pp. 1–13. DOI: 10.1186/s12943-018-0782-4. (Visited on 08/09/2024).

[193] Stevan R. Hubbard and Jeffrey H. Till. "Protein Tyrosine Kinase Structure and Function." In: (July 1, 2000). DOI: 10.1146/annurev.biochem.69.1.373. (Visited on 08/09/2024).

[194] Elodie Laine, Christian Auclair, and Luba Tchertanov. "Allosteric Communication across the Native and Mutated KIT Receptor Tyrosine Kinase." In: *PLoS Computational Biology* 8.8 (Aug. 2012). Ed. by Ruth Nussinov, e1002661. DOI: 10.1371/journal.pcbi.1002661. (Visited on 04/24/2024).

[195] Joseph Schlessinger and Mark A. Lemmon. "SH2 and PTB Domains in Tyrosine Kinase Signaling." In: *Science's STKE* 2003.191 (July 15, 2003), re12–re12. DOI: 10.1126/stke.2003.191.re12. (Visited on 08/09/2024).

[196] Peter van der Geer and Tony Pawson. "The PTB domain: a new protein module implicated in signal transduction." In: *Trends in Biochemical Sciences* 20.7 (July 1, 1995), pp. 277–280. DOI: 10.1016/S0968-0004(00)89043-X. (Visited on 08/30/2024).

[197] Melany J. Wagner et al. "Molecular Mechanisms of SH2- and PTB-Domain-Containing Proteins in Receptor Tyrosine Kinase Signaling." In: (Dec. 1, 2013). DOI: 10.1101/cshperspect.a008987. (Visited on 08/09/2024).

[198] Li Xing and Adrian Huang. "Bruton's TK Inhibitors: Structural Insights and Evolution of Clinical Candidates." In: *Future Medicinal*

Chemistry 6.6 (Apr. 1, 2014), pp. 675–695. DOI: 10.4155/fmc.14.24. (Visited on 08/09/2024).

[199] Marc Thiriet. "Cytoplasmic Protein Tyrosine Kinases." In: Jan. 1, 2012. DOI: 10.1007/978-1-4614-4370-4_4. (Visited on 08/09/2024).

[200] Shiqing Li et al. "Structural and Biochemical Evidence for an Autoinhibitory Role for Tyrosine 984 in the Juxtamembrane Region of the Insulin Receptor." In: *Journal of Biological Chemistry* 278.28 (July 2003), pp. 26007–26014. DOI: 10.1074/JBC.M302425200. (Visited on 03/04/2024).

[201] Priscila da Silva Figueiredo Celestino. "Oncogenic Mechanisms of Activation and Resistance of the type III Receptor Tyrosine Kinase family." PhD thesis. École normale supérieure de Cachan - ENS Cachan, June 26, 2015. (Visited on 08/09/2024).

[202] Rina Barouch-Bentov and Karsten Sauer. "Mechanisms of Drug-Resistance in Kinases." In: *Expert opinion on investigational drugs* 20.2 (Feb. 2011), pp. 153–208. DOI: 10.1517/13543784.2011.546344. (Visited on 01/20/2018).

[203] Julie Ledoux, Marina Botnari, and Luba Tchertanov. "Receptor Tyrosine Kinase KIT: Mutation-Induced Conformational Shift Promotes Alternative Allosteric Pockets." In: *Kinases and Phosphatases* 1.4 (Sept. 25, 2023). DOI: 10.3390/kinasesphosphatases1040014. (Visited on 08/09/2024).

[204] Lingfeng Chen et al. "Molecular basis for receptor tyrosine kinase A-loop tyrosine transphosphorylation." In: *Nature Chemical Biology* 16.3 (Mar. 2020), pp. 267–277. DOI: 10.1038/s41589-019-0455-7. (Visited on 08/09/2024).

[205] Yilin Meng and Benoît Roux. "Locking the Active Conformation of c-Src Kinase through the Phosphorylation of the Activation Loop." In: *Journal of Molecular Biology* 426.2 (Jan. 23, 2014), pp. 423–435. DOI: 10.1016/j.jmb.2013.10.001. (Visited on 08/09/2024).

[206] A. Ullrich et al. "Human epidermal growth factor receptor cDNA sequence and aberrant expression of the amplified gene in A431 epidermoid carcinoma cells." In: *Nature* 309.5967 (May 1984), pp. 418–425. DOI: 10.1038/309418a0. (Visited on 08/30/2024).

[207] Amit Arora and Eric M. Scholar. "Role of Tyrosine Kinase Inhibitors in Cancer Therapy." In: *Journal of Pharmacology and Experimental Therapeutics* 315.3 (Dec. 1, 2005), pp. 971–979. (Visited on 05/09/2024).

[208] Jorge Esteban-Villarrubia et al. "Tyrosine Kinase Receptors in Oncology." In: *International Journal of Molecular Sciences* 21.22 (Nov. 12, 2020), p. 8529. DOI: 10.3390/ijms21228529. (Visited on 08/09/2024).

[209] N. Sekhon, R. A. Kumbla, and M. Mita. "Chapter 1 - Current Trends in Cancer Therapy." In: *Cardio-Oncology*. Ed. by Roberta A. Gottlieb and Puja K. Mehta. Boston: Academic Press, Jan. 1, 2017, pp. 1–24. ISBN: 978-0-12-803547-4. DOI: 10.1016/B978-0-12-803547-4.00001-X. (Visited on 08/09/2024).

[210] Fleur Broekman, Elisa Giovannetti, and Godefridus J Peters. "Tyrosine kinase inhibitors: Multi-targeted or single-targeted?" In: *World Journal of Clinical Oncology* 2.2 (Feb. 10, 2011), pp. 80–93. DOI: 10.5306/wjco.v2.i2.80. (Visited on 08/09/2024).

[211] Steven A. Rosenzweig. "Acquired Resistance to Drugs Targeting Tyrosine Kinases." In: *Advances in cancer research* 138 (2018), pp. 71–98. DOI: 10.1016/bs.acr.2018.02.003. (Visited on 06/10/2019).

[212] Christine M. Lovly and Alice T. Shaw. "Molecular Pathways: Resistance to Kinase Inhibitors and Implications for Therapeutic Strategies." In: *Clinical Cancer Research* 20.9 (Apr. 30, 2014), pp. 2249–2256. DOI: 10.1158/1078-0432.CCR-13-1610. (Visited on 08/09/2024).

[213] Jeffrey A Engelman and Jeffrey Settleman. "Acquired resistance to tyrosine kinase inhibitors during cancer therapy." In: *Current Opinion in Genetics & Development* 18.1 (Feb. 2008), pp. 73–79. DOI: 10.1016/j.gde.2008.01.004. (Visited on 04/05/2024).

[214] Louise N. Johnson. "Protein kinase inhibitors: contributions from structure to clinical compounds." In: *Quarterly Reviews of Biophysics* 42.1 (Feb. 2009), pp. 1–40. DOI: 10.1017/S0033583508004745. (Visited on 08/30/2024).

[215] Amanda T.S. Albanaz et al. "Combating mutations in genetic disease and drug resistance: understanding molecular mechanisms to guide drug design." In: *Expert Opinion on Drug Discovery* 12.6 (June 3, 2017), pp. 553–563. DOI: 10.1080/17460441.2017.1322579. (Visited on 08/10/2024).

[216] Richard A. Ward et al. "Challenges and Opportunities in Cancer Drug Resistance." In: *Chemical Reviews* 121.6 (Mar. 24, 2021), pp. 3297–3351. DOI: 10.1021/acs.chemrev.0c00383. (Visited on 08/10/2024).

[217] Robert Roskoski. "Structure and regulation of Kit protein-tyrosine kinase—The stem cell factor receptor." In: *Biochemical and Biophysical Research Communications* 338.3 (Dec. 23, 2005), pp. 1307–1315. DOI: 10.1016/j.bbrc.2005.09.150. (Visited on 08/09/2024).

[218] Xiarong Shi et al. "Distinct cellular properties of oncogenic KIT receptor tyrosine kinase mutants enable alternative courses of cancer cell inhibition." In: *Proceedings of the National Academy of Sciences* 113.33 (Aug. 16, 2016), E4784–E4793. DOI: 10.1073/pnas.1610179113. (Visited on 08/09/2024).

[219] Priscila Da Silva Figueiredo Celestino Gomes et al. "Differential Effects of CSF-1R D802V and KIT D816V Homologous Mutations on Receptor Tertiary Structure and Allosteric Communication." In: *PLOS ONE* 9.5 (May 14, 2014), e97519. DOI: 10.1371/journal.pone.0097519. (Visited on 08/30/2024).

[220] Elodie Laine et al. "Mutation D816V Alters the Internal Structure and Dynamics of c-KIT Receptor Cytoplasmic Region: Implications for Dimerization and Activation Mechanisms." In: *PLoS Computational Biology* 7.6 (June 2011). Ed. by Gennady M. Verkhivker, e1002068. DOI: 10.1371/journal.pcbi.1002068. (Visited on 06/04/2024).

[221] Priscila Da Silva Figueiredo Celestino Gomes et al. "Insight on Mutation-Induced Resistance from Molecular Dynamics Simulations of the Native and Mutated CSF-1R and KIT." In: *PLOS ONE* 11.7 (July 28, 2016), e0160165. DOI: 10.1371/journal.pone.0160165. (Visited on 08/30/2024).

[222] Sean Caenepeel et al. "Motesanib inhibits Kit mutations associated with gastrointestinal stromal tumors." In: *Journal of Experimental & Clinical Cancer Research* 29.1 (July 15, 2010), p. 96. DOI: 10.1186/1756-9966-29-96. (Visited on 09/13/2024).

[223] S. Agarwal et al. "The activation loop tyrosine 823 is essential for the transforming capacity of the c-Kit oncogenic mutant D816V." In: *Oncogene* 34.35 (Aug. 2015), p. 4581. DOI: 10.1038/onc.2014.383. (Visited on 01/20/2018).

[224] Henrik Daub, Katja Specht, and Axel Ullrich. "Strategies to overcome resistance to targeted protein kinase inhibitors." In: *Nature Reviews Drug Discovery* 3.12 (Dec. 2004), pp. 1001–1010. DOI: 10.1038/nrd1579. (Visited on 08/09/2024).

[225] C. M. Wang et al. "Secondary resistance to imatinib in patients with gastrointestinal stromal tumors through an acquired KIT exon 17 mutation." In: *Molecular Medicine Reports* 2.3 (May 1, 2009), pp. 455–460. DOI: 10.3892/mmr_00000121. (Visited on 08/30/2024).

[226] Alessandra Maleddu et al. "Mechanisms of secondary resistance to tyrosine kinase inhibitors in gastrointestinal stromal tumours (Review)." In: *Oncology Reports* 21.6 (June 1, 2009), pp. 1359–1366. DOI: 10.3892/or_00000361. (Visited on 08/30/2024).

[227] Julhash U. Kazi et al. "Tyrosine 842 in the activation loop is required for full transformation by the oncogenic mutant FLT3-ITD." In: *Cellular and Molecular Life Sciences* 74.14 (Mar. 7, 2017), pp. 2679–2688. DOI: 10.1007/s00018-017-2494-0. (Visited on 08/09/2024).

[228] T Wakai et al. "Late resistance to imatinib therapy in a metastatic gastrointestinal stromal tumour is associated with a second KIT mutation." In: *British Journal of Cancer* 90.11 (June 1, 2004), pp. 2059–2061. DOI: 10.1038/sj.bjc.6601819. (Visited on 01/20/2018).

[229] Jonathan P. DiNitto et al. "Function of activation loop tyrosine phosphorylation in the mechanism of c-Kit auto-activation and its implication in sunitinib resistance." In: *The Journal of Biochemistry* 147.4 (Apr. 1, 2010), pp. 601–609. DOI: 10.1093/jb/mvq015. (Visited on 08/09/2024).

[230] Keiji Miyazawa. "Phosphorylation in the activation loop as the finishing touch in c-Kit activation." In: *The Journal of Biochemistry* 151.5 (May 1, 2012), pp. 457–459. DOI: 10.1093/jb/mvs031. (Visited on 08/10/2024).

[231] C. R. Antonescu et al. "Acquired Resistance to Imatinib in Gastrointestinal Stromal Tumor Occurs Through Secondary Gene Mutation." In: *Clinical Cancer Research* 11.11 (June 2005), pp. 4182–4190. DOI: 10.1158/1078-0432.CCR-04-2245. (Visited on 01/20/2018).

[232] Jie Zhao et al. "Flumatinib, a selective inhibitor of BCR-ABL / PDGFR / KIT, effectively overcomes drug resistance of certain KIT mutants." In: *Cancer Science* 105.1 (Jan. 2014), pp. 117–125. DOI: 10.1111/cas.12320. (Visited on 01/20/2018).

[233] Toshirou Nishida et al. "Secondary mutations in the kinase domain of the KIT gene are predominant in imatinib-resistant gastrointestinal stromal tumor." In: *Cancer Science* 99.4 (Apr. 2008), pp. 799–804. DOI: 10.1111/j.1349-7006.2008.00727.x. (Visited on 06/21/2019).

[234] Gianluca Spitaleri et al. "Inactivity of imatinib in gastrointestinal stromal tumors (GISTs) harboring a KIT activation-loop domain mutation (exon 17 mutation pN822K)." In: *OncoTargets and therapy* 8 (Aug. 18, 2015), pp. 1997–2003. DOI: 10.2147/OTT.S81558. (Visited on 06/10/2019).

[235] Morgan Huse and John Kuriyan. "The conformational plasticity of protein kinases." In: *Cell* 109.3 (May 2002), pp. 275–82. DOI: 10.1016/S0092-8674(02)00741-9. (Visited on 06/24/2019).

[236] Alexandr P. Kornev and Susan S. Taylor. "Defining the conserved internal architecture of a protein kinase." In: *Biochimica et Biophysica Acta (BBA) - Proteins and Proteomics* 1804.3 (Mar. 2010), pp. 440–444. DOI: 10.1016/J.BBAPAP.2009.10.017. (Visited on 10/06/2019).

[237] Alexandr P. Kornev, Susan S. Taylor, and Lynn F. Ten Eyck. "A helix scaffold for the assembly of active protein kinases." In: *Proceedings of the National Academy of Sciences* 105.38 (Sept. 23, 2008), pp. 14377–14382. DOI: 10.1073/pnas.0807988105. (Visited on 06/10/2019).

[238] Alexey Aleksandrov and Thomas Simonson. "Molecular Dynamics Simulations Show That Conformational Selection Governs the Binding Preferences of Imatinib for Several Tyrosine Kinases." In: *Journal of Biological Chemistry* 285.18 (Apr. 30, 2010), pp. 13807–13815. DOI: 10.1074/jbc.M110.109660. (Visited on 06/11/2019).

[239] Vladimir Alexei Morozov and Sylvie Lagaye. "Hepatitis C virus: Morphogenesis, infection and therapy." In: *World Journal of Hepatology* 10.2 (Feb. 27, 2018), pp. 186–212. DOI: 10.4254/wjh.v10.i2.186. (Visited on 08/22/2023).

[240] Stéphane Chevaliez and Jean-Michel Pawlotsky. "HCV Genome and Life Cycle." In: *Hepatitis C Viruses: Genomes and Molecular Biology*. Ed. by Seng-Lai Tan. Norfolk (UK): Horizon Bioscience, 2006. ISBN: 978-1-904933-20-5. (Visited on 04/29/2024).

[241] R Purcell. "The hepatitis C virus: Overview." In: *Hepatology* 26 (S3 1997), 11S–14S. DOI: 10.1002/hep.510260702. (Visited on 08/22/2023).

[242] Stephen M. Feinstone et al. "Transfusion-Associated Hepatitis Not Due to Viral Hepatitis Type A or B." In: *New England Journal of Medicine* 292.15 (Apr. 10, 1975), pp. 767–770. DOI: 10.1056/NEJM197504102921502. (Visited on 08/22/2023).

[243] Qui-Lim Choo et al. "Isolation of a cDNA cLone Derived from a Blood-Borne Non-A, Non-B Viral Hepatitis Genome." In: *Science* 244.4902 (Apr. 21, 1989), pp. 359–362. DOI: 10.1126/science.2523562. (Visited on 08/22/2023).

[244] Y K Shimizu et al. "Hepatitis C virus: Detection of intracellular virus particles by electron microscop." In: *Hepatology* 23.2 (1996), pp. 205–209. DOI: 10.1002/hep.510230202. (Visited on 08/22/2023).

[245] C. Giannini and C. Bréchot. "Hepatitis C virus biology." In: *Cell Death & Differentiation* 10.1 (Jan. 2003), S27–S38. DOI: 10.1038/sj.cdd.4401121. (Visited on 08/22/2023).

[246] Lize Cuypers et al. "Impact of HCV genotype on treatment regimens and drug resistance: a snapshot in time." In: *Reviews in Medical Virology* 26.6 (Nov. 2016), pp. 408–434. DOI: 10.1002/rmv.1895. (Visited on 05/04/2024).

[247] Johannes Vermehren and Christoph Sarrazin. "The role of resistance in HCV treatment." In: *Best Practice & Research Clinical Gastroenterology*. Chronic Hepatitis C: Diagnosis and Treatment 26.4 (Aug. 1, 2012), pp. 487–503. DOI: 10.1016/j.bpg.2012.09.011. (Visited on 05/04/2024).

[248] Gennadiy Koev and Warren Kati. "The emerging field of HCV drug resistance." In: *Expert Opinion on Investigational Drugs* 17.3 (Mar. 1, 2008), pp. 303–319. DOI: 10.1517/13543784.17.3.303. (Visited on 05/04/2024).

[249] Erik Lontok et al. "Hepatitis C virus drug resistance–associated substitutions: State of the art summary." In: *Hepatology* 62.5 (2015), pp. 1623–1632. DOI: 10.1002/hep.27934. (Visited on 05/04/2024).

[250] Darius Moradpour and François Penin. "Hepatitis C Virus Proteins: From Structure to Function." In: *Hepatitis C Virus: From Molecular Virology to Antiviral Therapy*. Ed. by Ralf Bartenschlager. Current Topics in Microbiology and Immunology. Berlin, Heidelberg: Springer, 2013, pp. 113–142. ISBN: 978-3-642-27340-7. DOI: 10.1007/978-3-642-27340-7_5. (Visited on 08/22/2023).

[251] Kevin D. Raney et al. "Hepatitis C Virus Non-structural Protein 3 (HCV NS3): A Multifunctional Antiviral Target *." In: *Journal of Biological Chemistry* 285.30 (July 23, 2010), pp. 22725–22731. DOI: 10.1074/jbc.R110.125294. (Visited on 04/30/2024).

[252] Chao Lin. "HCV NS3-4A Serine Protease." In: *Hepatitis C Viruses: Genomes and Molecular Biology*. Ed. by Seng-Lai Tan. Norfolk (UK): Horizon Bioscience, 2006. ISBN: 978-1-904933-20-5. (Visited on 05/05/2024).

[253] R. Bartenschlager. "The NS3/4A proteinase of the hepatitis C virus: unravelling structure and function of an unusual enzyme and a prime target for antiviral therapy." In: *Journal of Viral Hepatitis* 6.3 (May 1999), pp. 165–181. DOI: 10.1046/j.1365-2893.1999.00152.x. (Visited on 05/05/2024).

[254] Fiona McPhee et al. "Resistance Analysis of the Hepatitis C Virus NS3 Protease Inhibitor Asunaprevir." In: *Antimicrobial Agents and Chemotherapy* 56.7 (June 14, 2012), pp. 3670–3681. DOI: 10.1128/aac.00308-12. (Visited on 05/05/2024).

[255] Samantha J. Shepherd et al. "Prevalence of HCV NS3 pre-treatment resistance associated amino acid variants within a Scottish cohort." In: *Journal of Clinical Virology* 65 (Apr. 1, 2015), pp. 50–53. DOI: 10.1016/j.jcv.2015.02.005. (Visited on 05/05/2024).

[256] Tina Ruggiero et al. "Predominance of hepatitis C virus Q80K among NS3 baseline-resistance-associated amino acid variants in direct-antiviral-agent-naïve patients with chronic hepatitis: single-centre experience." In: *Archives of Virology* 160.11 (Nov. 1, 2015), pp. 2881–2885. DOI: 10.1007/s00705-015-2563-3. (Visited on 05/05/2024).

[257] Ruihong Wu et al. "Computational analysis of naturally occurring resistance-associated substitutions in genes NS3, NS5A, and NS5B among 86 subtypes of hepatitis C virus worldwide." In: *Infection and Drug Resistance* 12 (Sept. 19, 2019), pp. 2987–3015. DOI: 10.2147/IDR.S218584. (Visited on 05/05/2024).

[258] Rosemary M. McCloskey et al. "Global Origin and Transmission of Hepatitis C Virus Nonstructural Protein 3 Q80K Polymorphism." In: *The Journal of Infectious Diseases* 211.8 (Apr. 15, 2015), pp. 1288–1295. DOI: 10.1093/infdis/jiu613. (Visited on 08/22/2023).

[259] Allan Peres-da Silva et al. "Effects of the Q80K Polymorphism on the Physicochemical Properties of Hepatitis C Virus Subtype 1a NS3 Protease." In: *Viruses* 11.8 (July 30, 2019), p. 691. DOI: 10.3390/v11080691. (Visited on 08/22/2023).

[260] Aruna Sampath and R. Padmanabhan. "Molecular targets for flavivirus drug discovery." In: *Antiviral Research* 81.1 (Jan. 2009), pp. 6–15. DOI: 10.1016/j.antiviral.2008.08.004. (Visited on 03/06/2024).

[261] Auda A. Eltahla et al. "Dynamic evolution of hepatitis C virus resistance-associated substitutions in the absence of antiviral treatment." In: *Scientific Reports* 7.1 (Jan. 31, 2017), p. 41719. DOI: 10.1038/srep41719. (Visited on 05/06/2024).

[262] Christophe Combet et al. "euHCVdb: the European hepatitis C virus database." In: *Nucleic Acids Research* 35 (Database issue Jan. 2007), pp. D363–366. DOI: 10.1093/nar/gkl970. (Visited on 05/06/2024).

[263] Keith P. Romano et al. "Drug resistance against HCV NS3/4A inhibitors is defined by the balance of substrate recognition versus inhibitor binding." In: *Proceedings of the National Academy of Sciences of the United States of America* 107.49 (Dec. 7, 2010), pp. 20986–20991. DOI: 10.1073/pnas.1006370107. (Visited on 07/31/2024).

[264] Lavi S. Bigman and Yaakov Levy. "Stability Effects of Protein Mutations: The Role of Long-Range Contacts." In: *The Journal of Physical Chemistry. B* 122.49 (Dec. 13, 2018), pp. 11450–11459. DOI: 10.1021/acs.jpcb.8b07379.

[265] Esteban Domingo, Julie Sheldon, and Celia Perales. "Viral Quasispecies Evolution." In: *Microbiology and Molecular Biology Reviews : MMBR* 76.2 (June 2012), pp. 159–216. DOI: 10.1128/MMBR.05023-11. (Visited on 07/31/2024).

[266] Libin Rong et al. "Rapid emergence of protease inhibitor resistance in hepatitis C virus." In: *Science translational medicine* 2.30 (May 5, 2010), 30ra32. DOI: 10.1126/scitranslmed.3000544. (Visited on 07/31/2024).

[267] Masato Ogishi et al. "Deconvoluting the Composition of Low-Frequency Hepatitis C Viral Quasispecies: Comparison of Genotypes and NS3 Resistance-Associated Variants between HCV/HIV Coinfected Hemophiliacs and HCV Monoinfected Patients in Japan." In: *PLOS ONE* 10.3 (Mar. 6, 2015), e0119145. DOI: 10.1371/journal.pone.0119145. (Visited on 08/31/2024).

[268] Long V. Pham et al. "HCV genotype 1-6 NS3 residue 80 substitutions impact protease inhibitor activity and promote viral escape." In: *Journal of Hepatology* 70.3 (Mar. 2019), pp. 388–397. DOI: 10.1016/j.jhep.2018.10.031. (Visited on 08/22/2023).

[269] M. V. Lin et al. "Hepatitis C virus NS3 mutations in haemophiliacs." In: *Haemophilia* 20.5 (2014), pp. 659–665. DOI: 10.1111/hae.12420. (Visited on 07/31/2024).

[270] Stefania Paolucci et al. "Naturally occurring mutations to HCV protease inhibitors in treatment-naïve patients." In: *Virology Journal* 9.1 (Oct. 24, 2012), pp. 1–8. DOI: 10.1186/1743-422X-9-245. (Visited on 07/31/2024).

[271] Christoph Welsch et al. "Hepatitis C virus variants resistant to macrocyclic NS3-4A inhibitors subvert IFN-β induction by efficient MAVS cleavage." In: *Journal of Hepatology* 62.4 (Apr. 2015), pp. 779–784. DOI: 10.1016/j.jhep.2014.11.009. (Visited on 08/06/2023).

[272] Georg Dultz et al. "Extended interaction networks with HCV protease NS3-4A substrates explain the lack of adaptive capability against protease inhibitors." In: *The Journal of Biological Chemistry* 295.40 (Oct. 2, 2020), pp. 13862–13874. DOI: 10.1074/jbc.RA120.013898. (Visited on 05/06/2024).

[273] MinKyung Yi et al. "Mutations conferring resistance to SCH6, a novel hepatitis C virus NS3/4A protease inhibitor. Reduced RNA replication fitness and partial rescue by second-site mutations." In: *The Journal of Biological Chemistry* 281.12 (Mar. 24, 2006), pp. 8205–8215. DOI: 10.1074/jbc.M510246200. (Visited on 08/06/2023).

[274] Christoph Welsch et al. "Peptidomimetic escape mechanisms arise via genetic diversity in the ligand-binding site of the hepatitis C virus NS3/4A serine protease." In: *Gastroenterology* 142.3 (Mar. 2012), pp. 654–663. DOI: 10.1053/j.gastro.2011.11.035. (Visited on 05/06/2024).

[275] Burton H. Bloom. "Space/time trade-offs in hash coding with allowable errors." In: *Communications of the ACM* 13.7 (July 1, 1970), pp. 422–426. DOI: 10.1145/362686.362692. (Visited on 06/20/2022).

[276] Thomas H. Cormen et al. *Introduction to Algorithms, fourth edition*. MIT Press, Apr. 5, 2022. 1313 pp. ISBN: 978-0-262-36750-9. (Visited on 08/17/2024).

[277] Robert Sedgewick and Kevin Wayne. *Algorithms*. Addison-Wesley Professional, 2011. 968 pp. ISBN: 978-0-321-57351-3. (Visited on 08/17/2024).

[278] N. Yazdani and P.S. Min. "Prefix trees: new efficient data structures for matching strings of different lengths." In: *Proceedings 2001 International Database Engineering and Applications Symposium*. Proceedings 2001 International Database Engineering and Applications Symposium. July 2001, pp. 76–85. DOI: 10.1109/IDEAS.2001.938073. (Visited on 08/17/2024).

[279] Prashant Pandey et al. "deBGR: an efficient and near-exact representation of the weighted de Bruijn graph." In: *Bioinformatics* 33.14 (July 15, 2017), pp. i133–i141. DOI: 10.1093/bioinformatics/btx261. (Visited on 08/29/2024).

[280] Bin Fan et al. "Cuckoo Filter: Practically Better Than Bloom." In: *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*. CoNEXT '14. New York, NY, USA: Association for Computing Machinery, Dec. 2, 2014, pp. 75–88. ISBN: 978-1-4503-3279-8. DOI: 10.1145/2674005.2674994. (Visited on 08/24/2023).

[281] Michael A. Bender et al. "Don't thrash: how to cache your hash on flash." In: *Proceedings of the VLDB Endowment* 5.11 (July 2012), pp. 1627–1637. DOI: 10.14778/2350229.2350275. (Visited on 08/24/2023).

[282] Fan Deng and Davood Rafiei. "Approximately detecting duplicates for streaming data using stable bloom filters." In: *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*. SIGMOD '06. New York, NY, USA: Association for Computing Machinery, June 27, 2006, pp. 25–36. ISBN: 978-1-59593-434-5. DOI: 10.1145/1142473.1142477. (Visited on 08/29/2024).

[283] Yang Liu, Wenji Chen, and Yong Guan. "Near-optimal approximate membership query over time-decaying windows." In: *2013 Proceedings IEEE INFOCOM*. 2013 Proceedings IEEE INFOCOM. Apr. 2013, pp. 1447–1455. DOI: 10.1109/INFCOM.2013.6566939. (Visited on 08/29/2024).

[284] Sourav Dutta, Ankur Narang, and Suman K. Bera. "Streaming quotient filter: a near optimal approximate duplicate detection approach for data streams." In: *Proc. VLDB Endow.* 6.8 (June 1, 2013), pp. 589–600. DOI: 10.14778/2536354.2536359. (Visited on 08/29/2024).

[285] Biplob Debnath et al. "BloomFlash: Bloom Filter on Flash-Based Storage." In: *2011 31st International Conference on Distributed Computing Systems*. 2011 31st International Conference on Distributed Computing Systems. June 2011, pp. 635–644. DOI: 10.1109/ICDCS.2011.44. (Visited on 08/29/2024).

[286] Rasmus Pagh and Flemming Friche Rodler. "Cuckoo hashing." In: *Journal of Algorithms* 51.2 (May 1, 2004), pp. 122–144. DOI: 10.1016/j.jalgor.2003.12.002. (Visited on 08/24/2023).

[287] Jan Grashöfer, Florian Jacob, and Hannes Hartenstein. "Towards Application of Cuckoo Filters in Network Security Monitoring." In: *2018 14th International Conference on Network and Service Management (CNSM)*. 2018 14th International Conference on Network and Service Management (CNSM). Nov. 2018, pp. 373–377. (Visited on 09/12/2024).

[288] Antonio Sérgio Cruz Gaia et al. "NGSReadsTreatment – A Cuckoo Filter-based Tool for Removing Duplicate Reads in NGS Data." In: *Scientific Reports* 9.1 (Aug. 12, 2019), p. 11681. DOI: 10.1038/s41598-019-48242-w. (Visited on 08/21/2024).

[289] Prashant Pandey et al. "A General-Purpose Counting Filter: Making Every Bit Count." In: *Proceedings of the 2017 ACM International Conference on Management of Data*. SIGMOD/PODS'17: International Conference on Management of Data. Chicago Illinois USA: ACM, May 9, 2017, pp. 775–787. ISBN: 978-1-4503-4197-4. DOI: 10.1145/3035918.3035963. (Visited on 08/23/2023).

[290] R. González and V. Mäkinen. "PRACTICAL IMPLEMENTATION OF RANK AND SELECT QUERIES." In: 2005. (Visited on 08/29/2024).

[291] Brad Solomon and Carl Kingsford. "Fast search of thousands of short-read sequencing experiments." In: *Nature Biotechnology* 34.3 (Mar. 2016), pp. 300–302. DOI: 10.1038/nbt.3442. (Visited on 02/10/2020).

[292] Christiam Camacho et al. "BLAST+: architecture and applications." In: *BMC Bioinformatics* 10 (Dec. 15, 2009), p. 421. DOI: 10.1186/1471-2105-10-421. (Visited on 08/18/2024).

[293] Alexander Dobin et al. "STAR: ultrafast universal RNA-seq aligner." In: *Bioinformatics* 29.1 (Jan. 2013), pp. 15–21. DOI: 10.1093/bioinformatics/bts635. (Visited on 08/18/2024).

[294] Brad Solomon and Carl Kingsford. "Improved Search of Large Transcriptomic Sequencing Databases Using Split Sequence Bloom Trees." In: *21st Annual International Conference on Research in Com-*

putational Molecular Biology - Volume 10229. RECOMB 2017. Berlin, Heidelberg: Springer-Verlag, May 3, 2017, pp. 257–271. ISBN: 978-3-319-56969-7. DOI: 10.1007/978-3-319-56970-3_16. (Visited on 08/18/2024).

[295] Chen Sun et al. "AllSome Sequence Bloom Trees." In: *Journal of Computational Biology* 25.5 (May 2018), pp. 467–479. DOI: 10.1089/cmb.2017.0258. (Visited on 08/18/2024).

[296] Rajeev Raman, Venkatesh Raman, and Srinivasa Rao Satti. "Succinct indexable dictionaries with applications to encoding k -ary trees, prefix sums and multisets." In: *ACM Transactions on Algorithms* 3.4 (Nov. 2007), p. 43. DOI: 10.1145/1290672.1290680. (Visited on 08/29/2024).

[297] Robert S Harris and Paul Medvedev. "Improved representation of sequence bloom trees." In: *Bioinformatics* 36.3 (Aug. 22, 2019), pp. 721–727. DOI: 10.1093/bioinformatics/btz662. (Visited on 07/25/2021).

[298] Prashant Pandey et al. "Mantis: A Fast, Small, and Exact Large-Scale Sequence-Search Index." In: *Cell Systems* 7.2 (Aug. 22, 2018), 201–207.e4. DOI: 10.1016/j.cels.2018.05.021. (Visited on 07/25/2021).

[299] Zamin Iqbal et al. "De novo assembly and genotyping of variants using colored de Bruijn graphs." In: *Nature genetics* 44.2 (Jan. 8, 2012), pp. 226–232. DOI: 10.1038/ng.1028. (Visited on 08/03/2021).

[300] Camille Marchet et al. "Data structures based on k-mers for querying large collections of sequencing data sets." In: *Genome Research* 31.1 (Jan. 1, 2021), pp. 1–12. DOI: 10.1101/gr.260604.119. (Visited on 07/06/2021).

[301] Prashant Pandey et al. "Squeakr: an exact and approximate k-mer counting system." In: *Bioinformatics* 34.4 (Feb. 15, 2018), pp. 568–575. DOI: 10.1093/bioinformatics/btx636. (Visited on 03/30/2022).

[302] Phelim Bradley et al. "Ultrafast search of all deposited bacterial and viral genomic data." In: *Nature Biotechnology* 37.2 (Feb. 2019),

pp. 152–159. DOI: 10.1038/s41587-018-0010-1. (Visited on 07/25/2021).

[303] Timo Bingmann et al. "COBS: A Compact Bit-Sliced Signature Index." In: Oct. 3, 2019. DOI: 10.1007/978-3-030-32686-9_21. (Visited on 08/19/2024).

[304] Temesgen Hailemariam Dadi et al. "DREAM-Yara: an exact read mapper for very large databases with short update time." In: *Bioinformatics* 34.17 (Sept. 1, 2018), pp. i766–i772. DOI: 10.1093/bioinformatics/bty567. (Visited on 12/06/2022).

[305] Bruno Codenotti et al. "Approximation algorithms for a hierarchically structured bin packing problem." In: *Information Processing Letters* 89.5 (Mar. 16, 2004), pp. 215–221. DOI: 10.1016/j.ipl.2003.12.001. (Visited on 08/19/2024).

[306] Eric W Sayers et al. "GenBank." In: *Nucleic Acids Research* 47 (D1 Jan. 8, 2019), pp. D94–D99. DOI: 10.1093/nar/gky989. (Visited on 08/29/2024).

[307] Conrad L Schoch et al. "NCBI Taxonomy: a comprehensive update on curation, resources and tools." In: *Database* 2020 (Jan. 1, 2020), baaa062. DOI: 10.1093/database/baaa062. (Visited on 08/29/2024).

[308] Enrico Siragusa. "Approximate string matching for high-throughput sequencing." In: (2015). DOI: 10.17169/refubium-15562. (Visited on 08/19/2024).

[309] Santiago Marco-Sola et al. "The GEM mapper: fast, accurate and versatile alignment by filtration." In: *Nature Methods* 9.12 (Dec. 2012), pp. 1185–1188. DOI: 10.1038/nmeth.2221. (Visited on 08/19/2024).

[310] Téo Lemane et al. "kmtricks: efficient and flexible construction of Bloom filters for large sequencing data collections." In: *Bioinformatics Advances* 2.1 (Jan. 1, 2022), vbac029. DOI: 10.1093/bioadv/vbac029. (Visited on 06/20/2022).

[311] Rayan Chikhi, Antoine Limasset, and Paul Medvedev. "Compacting de Bruijn graphs from sequencing data quickly and in low memory." In: *Bioinformatics* 32.12 (June 15, 2016), pp. i201–i208. DOI: 10.1093/bioinformatics/btw279. (Visited on 03/30/2022).

[312] Camille Marchet et al. "REINDEER: efficient indexing of k-mer presence and abundance in sequencing datasets." In: *Bioinformatics* 36 (Supplement_1 July 1, 2020), pp. i177–i185. DOI: 10.1093/bioinformatics/btaa487. (Visited on 07/25/2021).

[313] Lianming Du et al. "Pyfastx: a robust Python package for fast random access to sequences from plain and gzipped FASTA/Q files." In: *Briefings in Bioinformatics* 22.4 (July 1, 2021). DOI: 10.1093/bib/bbaa368. (Visited on 07/25/2021).

[314] Alexander Gress et al. "StructMAn: annotation of single-nucleotide polymorphisms in the structural context." In: *Nucleic Acids Research* 44 (Web Server issue July 8, 2016), W463–W468. DOI: 10.1093/nar/gkw364. (Visited on 04/04/2022).

[315] Fumito Yamaguchi and Hiroaki Nishi. "Hardware-based hash functions for network applications." In: 2013 19th IEEE International Conference on Networks (ICON). Dec. 2013, pp. 1–6. DOI: 10.1109/ICON.2013.6781990. (Visited on 08/19/2024).

[316] César Estébanez et al. "Performance of the most common non-cryptographic hash functions." In: *Software: Practice and Experience* 44.6 (2014), pp. 681–698. DOI: 10.1002/spe.2179. (Visited on 08/19/2024).

[317] Yago Saez et al. "Evolutionary hash functions for specific domains." In: *Applied Soft Computing* 78 (May 1, 2019), pp. 58–69. DOI: 10.1016/j.asoc.2019.02.014. (Visited on 08/19/2024).

[318] Alistair Miles et al. *zarr-developers/zarr-python: v2.4.0*. Jan. 11, 2020. DOI: 10.5281/zenodo.3773450. (Visited on 06/29/2021).

[319] K. Masui et al. "A compression scheme for radio data in high performance computing." In: *Astronomy and Computing* 12 (Sept. 1, 2015), pp. 181–190. DOI: 10.1016/j.ascom.2015.07.002. (Visited on 08/29/2024).

[320] Nuala A. O'Leary et al. "Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation." In: *Nucleic Acids Research* 44 (Database issue Jan. 4, 2016), pp. D733–D745. DOI: 10.1093/nar/gkv1189. (Visited on 04/07/2022).

[321] Peter E.M. Taschner and Johan T. den Dunnen. "Describing structural changes by extending HGVS sequence variation nomenclature." In: *Human Mutation* 32.5 (2011), pp. 507–511. DOI: 10.1002/humu.21427. (Visited on 08/26/2024).

[322] Johan T. den Dunnen. "Describing Sequence Variants Using HGVS Nomenclature." In: *Genotyping: Methods and Protocols*. Ed. by Stefan J. White and Stuart Cantsilieris. New York, NY: Springer, 2017, pp. 243–251. ISBN: 978-1-4939-6442-0. DOI: 10.1007/978-1-4939-6442-0_17. (Visited on 08/26/2024).

[323] Alexander Gress et al. "d-StructMAn: Containerized structural annotation on the scale from genetic variants to whole proteomes." In: *GigaScience* 11 (Jan. 1, 2022), giac086. DOI: 10.1093/gigascience/giac086. (Visited on 08/26/2024).

[324] Heidi L. Rehm et al. "GA4GH: International policies and standards for data sharing across genomic research and healthcare." In: *Cell Genomics* 1.2 (Nov. 10, 2021), p. 100029. DOI: 10.1016/j.xgen.2021.100029. (Visited on 08/26/2024).

[325] Marc Fiume et al. "Federated discovery and sharing of genomic data using Beacons." In: *Nature Biotechnology* 37.3 (Mar. 2019), pp. 220–224. DOI: 10.1038/s41587-019-0046-x. (Visited on 08/26/2024).

[326] Jordi Rambla et al. "Beacon v2 and Beacon networks: A "lingua franca" for federated data discovery in biomedical genomics, and beyond." In: *Human Mutation* 43.6 (June 2022), pp. 791–799. DOI: 10.1002/humu.24369. (Visited on 08/26/2024).

[327] The Galaxy Community. "The Galaxy platform for accessible, reproducible, and collaborative data analyses: 2024 update." In: *Nucleic Acids Research* 52 (W1 July 5, 2024), W83–W94. DOI: 10.1093/nar/gkae410. (Visited on 09/02/2024).

[328] Björn Grüning et al. "Bioconda: sustainable and comprehensive software distribution for the life sciences." In: *Nature Methods* 15.7 (July 2018), pp. 475–476. DOI: 10.1038/s41592-018-0046-7. (Visited on 09/02/2024).

[329] Daniel Blankenberg et al. "Dissemination of scientific software with Galaxy ToolShed." In: *Genome Biology* 15.2 (Feb. 20, 2014), p. 403. DOI: 10.1186/gb4161. (Visited on 09/02/2024).