BIOPYVIEW: A BIOPYTHON-BASED PLATFORM FOR SEQUENCE ANALYSIS

by

Ömer Faruk Çavaş

B.S., Industrial Engineering, Boğaziçi University, 2020

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering

Boğaziçi University

2025

ACKNOWLEDGEMENTS

First and foremost, I express my gratitude and praise to Allah, the Owner of all things, for every accomplishment, big or small. It is He who has granted me the abilities, patience, and opportunities necessary to complete this work.

I am also deeply grateful to my family for their unwavering support throughout my entire education and particularly during this master's program. Their belief in me and their constant encouragement have been invaluable in helping me overcome challenges and stay focused on my goals.

I would like to express my grateful thanks to my advisor, Arzucan Özgür, for her invaluable guidance and constructive feedback, throughout this journey. Her expertise and encouragement have been instrumental in shaping this research and helping me deal with the difficulties of this work.

I am also thankful to Boğaziçi University for providing a supportive academic environment and essential technical resources, including access to journals, libraries, and free development tools, all of which were helpful to my research.

ABSTRACT

BIOPYVIEW: A BIOPYTHON-BASED PLATFORM FOR SEQUENCE ANALYSIS

Advancements in sequencing technologies have increased the demand for bioinformatics tools to process sequence and alignment data. Tools that support multiple sequence alignment are essential for tasks like evolutionary studies, motif discovery, and phylogenetics. However, many existing tools suffer from limitations such as restricted file format support, inadequate annotation capabilities, underdeveloped motif analysis, and inefficiency in handling large datasets. Furthermore, complex user interfaces often significantly hinder usability.

This thesis proposes the development of a modern, user-friendly, and cross-platform sequence analysis platform built on Biopython, a versatile and actively maintained bioinformatics library. The tool integrates essential features such as sequence viewing and editing, multiple sequence alignment, annotation management, motif analysis, and phylogenetic tree support while addressing key gaps in existing software. Key innovations include enhanced file format compatibility, an integrated annotation viewer and editor, advanced pattern search functionality, and memory-efficient handling of large datasets and an indexing mechanism that supports working with extremely large files.

By addressing limitations in existing tools and leveraging Biopython's robust framework, this work presents a scalable and versatile platform for biological sequence analysis. The proposed tool meets the growing needs of researchers in analyzing and interpreting large and complex biological datasets, paving the way for future advancements in bioinformatics.

ÖZET

BIOPYVIEW: DİZİ ANALİZİ İÇİN BIOPYTHON TABANLI BİR PLATFORM

Dizileme teknolojilerindeki ilerlemeler, dizi ve hizalama verilerini işlemek için biyoenformatik araçlarına olan talebi artırmıştır. Çoklu dizi hizalamasını destekleyen araçlar, evrimsel çalışmalar, motif keşfi ve filogenetik gibi görevler için hayati öneme sahiptir. Ancak, mevcut araçların birçoğu, sınırlı dosya formatı desteği, yetersiz anotasyon yetenekleri, az gelişmiş motif analizi ve büyük veri setlerini işleme konusundaki verimsizlik gibi sınırlamalara sahiptir. Ayrıca, karmaşık kullanıcı arayüzleri genellikle kullanılabilirliği önemli ölçüde engeller.

Bu tez, Biopython üzerine inşa edilmiş, modern, kullanıcı dostu ve çoklu platform desteği sunan bir dizi analiz platformunun geliştirilmesini önermektedir. Biopython, çok yönlü ve aktif olarak geliştirilen bir biyoenformatik kütüphanesidir. Önerilen araç, dizi görüntüleme ve düzenleme, çoklu dizi hizalaması, anotasyon yönetimi, motif analizi ve filogenetik ağaç desteği gibi temel özellikleri entegre ederek mevcut yazılımlardaki önemli eksiklikleri ele almayı amaçlamaktadır. Önemli yenilikler arasında gelişmiş dosya formatı uyumluluğu, entegre bir anotasyon görüntüleyici ve düzenleyici, ileri düzey desen arama işlevselliği, büyük veri setlerini hafıza verimliliğiyle işleme ve çok büyük boyutlu dosyaları destekleyen bir indeksleme mekanizması bulunmaktadır.

Mevcut araçlardaki sınırlamaları ele alarak ve Biopython'un güçlü altyapısından yararlanarak, bu çalışma biyolojik dizi analizi için ölçeklenebilir ve çok yönlü bir platform sunmaktadır. Önerilen araç, büyük ve karmaşık biyolojik veri setlerini analiz etmek ve yorumlamak isteyen araştırmacıların artan ihtiyaçlarını karşılayarak biyoenformatikte gelecekteki ilerlemelere zemin hazırlamaktadır.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . iii
ABSTRACT . . . iv
ÖZET . . . v
LIST OF FIGURES . . . x
LIST OF TABLES . . . xiii
LIST OF ACRONYMS/ABBREVIATIONS . . . xiv
1. INTRODUCTION . . . 1
2. LITERATURE REVIEW . . . 5
2.1. File Operations . . . 7
2.2. Sequence Operations . . . 8
2.3. Alignment . . . 9
2.4. Annotations . . . 11
2.5. Phylogenetic Tree Construction and Visualization . . . 12
2.6. Motif Search . . . 13
2.7. Analysis and Specialized Features . . . 14
2.8. Handling Large Files . . . 15
2.9. Cross-Platform Compatibility . . . 16
2.10. Critical Analysis and Identified Gaps . . . 16
3. METHODOLOGY . . . 19
3.1. Development Framework and Tools . . . 19
3.1.1. Biopython . . . 19
3.1.2. Tkinter . . . 20
3.1.3. Additional Tools and Libraries . . . 21
3.2. Platform Overview . . . 22
3.2.1. File . . . 22
3.2.1.1. Open Sequence File . . . 22
3.2.1.2. Open Sequence File as Index . . . 23
3.2.1.3. Open Alignment File . . . 25

3.2.1.4. Save File . . . 25
3.2.1.5. Convert Sequence File Format . . . 26
3.2.1.6. Convert Alignment File Format . . . 27
3.2.1.7. Settings . . . 27
3.2.2. Edit . . . 29
3.2.2.1. Editor Mode (Keyboard) . . . 29
3.2.2.2. Undo-Redo . . . 29
3.2.2.3. Copy-Paste . . . 29
3.2.2.4. Delete-Insert . . . 30
3.2.3. Select . . . 31
3.2.4. View . . . 32
3.2.5. Sequence . . . 32
3.2.5.1. Add New Sequence . . . 32
3.2.5.2. Concatenate Selected . . . 33
3.2.5.3. Complement Selected . . . 33
3.2.5.4. Reverse Complement Selected: . . . 34
3.2.5.5. Transcribe Selected . . . 34
3.2.5.6. Translate Selected . . . 34
3.2.6. Alignment . . . 35
3.2.6.1. Switch Alignment . . . 35
3.2.6.2. Multiple Sequence Alignment . . . 35
3.2.6.3. Pairwise Alignment . . . 36
3.2.6.4. Manual Alignment . . . 36
3.2.7. Annotation . . . 37
3.2.7.1. Information Tab . . . 38
3.2.7.2. Annotations Tab . . . 39
3.2.7.3. Features Tab . . . 40
3.2.7.4. Letter Annotation Tab . . . 41
3.2.7.5. Database Cross-References Tab . . . 42
3.2.8. Phylogenetic Tree . . . 43
3.2.8.1. Open Tree . . . 43

3.2.8.2. Infer Tree 44
3.2.8.3. Convert File Format 44
3.2.9. Motif 44
3.2.9.1. Create Motif 45
3.2.9.2. Read Motif 46
3.2.9.3. Search Options 49
3.2.10. Other Tools: 50
4. RESULTS AND EVALUATION 53
4.1. Functionality Comparisons with Existing Alternatives 53
4.1.1. Built on Biopython 53
4.1.2. Wide File Format Support 54
4.1.2.1. Reading Files with Multiple MSAs 55
4.1.3. Annotation Viewer and Editor 57
4.1.4. Extensive Pattern Search 60
4.1.5. Index Mode: Handling Very Large Files 60
4.1.6. Cross-Platform Compatibility 61
4.2. Memory Usage Evaluation 61
4.2.1. Testing Methodology 62
4.2.2. Test Results 64
4.3. Summary of Comparison 67
4.4. Limitations and Potential Improvements 68
5. CONCLUSION 71
5.1. Key Achievements 71
5.1.1. Wide File Format Support 71
5.1.2. Advanced Annotation Support 72
5.1.3. Enhanced Pattern Search 72
5.1.4. Index Mode: Handling Large Datasets 72
5.1.5. Memory Efficiency 73
5.2. Limitations 73
5.3. Future Work 73
5.4. Final Remarks 74

REFERENCES 75
APPENDIX A: ANNOTATION WINDOWS 80

LIST OF FIGURES

Figure 3.1. Sequence file dialog window. 23

Figure 3.2. Sequence viewer window. 24

Figure 3.3. Alignment selector. 26

Figure 3.4. Save window. 26

Figure 3.5. General settings. 27

Figure 3.6. Alignment tools settings. 28

Figure 3.7. Tree tools settings. 29

Figure 3.8. Other tools settings. 30

Figure 3.9. Example selection. 31

Figure 3.10. Background color mode. 33

Figure 3.11. Translate window. 34

Figure 3.12. Muscle tool window. 36

Figure 3.13. Muscle tool progress window. 36

Figure 3.14. An example FASTA file. 38

Figure 3.15. An example GenBank file. 39

Figure 3.16. Information tab. 40

Figure 3.17. Annotation tab. 41

Figure 3.18. The app's features tab. 42

Figure 3.19. Editing location in features tab. 43

Figure 3.20. Letter annotation tab. 44

Figure 3.21. Database cross-references tab. 45

Figure 3.22. FigTree window. 45

Figure 3.23. FastTree window. 46

Figure 3.24. Tree convert window. 47

Figure 3.25. Motif creation using instances. 48

Figure 3.26. Motif creation using count matrix. 50

Figure 3.27. Count matrix. 50

Figure 3.28. PWM matrix. 51

Figure 3.29. PSSM matrix. 51

Figure 3.30. Minimized search window. 52

Figure 3.31. Motif search options. 52

Figure 4.1. Annotations windows. 58

Figure A.1. The annotations in the GenBank file. 80

Figure A.2. The annotations displayed in the app's annotation tab. 80

Figure A.3. Reference annotations in the GenBank file. 81

Figure A.4. Reference annotations in the app's annotation tab. 81

Figure A.5. Feature table elements in the GenBank file. 82

Figure A.6. Feature table elements in the app. 83

Figure A.7. CDS feature in the GenBank file. 84

Figure A.8. CDS feature in the app. 84

LIST OF TABLES

Table 4.1. File format support comparison table. 55

Table 4.2. File formats only available in our app. 56

Table 4.3. File format support comparison table for annotations-rich formats. 59

Table 4.4. Index mode results. 61

Table 4.5. Test cases for DNA Sequences 63

Table 4.6. Memory test results. 64

Table 4.7. Feature comparison table. 68

LIST OF ACRONYMS/ABBREVIATIONS

EMBL-EBI	European Molecular Biology Laboratory - European Bioinformatics Institute
GUI	Graphical User Interface
IUPAC	International Union of Pure and Applied Chemistry
MSA	Multiple Sequence Alignment
NCBI	National Center for Biotechnology Information
PCA	Principal Component Analysis
PDB	Protein Data Bank
PSSM	Position-Specific Scoring Matrix
PWM	Position Weight Matrix
RSS	Resident Set Size
UI	User Interface

1. INTRODUCTION

The analysis of DNA, RNA, and protein sequences is central to modern biological research, enabling scientists to reveal the mechanisms of life. Bioinformatics applications have revolutionized the biological research by facilitating the analysis of molecular data. Among the most indispensable techniques in bioinformatics are Multiple Sequence Alignments (MSAs), which allow the comparison of sequences to identify evolutionary relationships, conserved regions, and functional motifs. Furthermore, MSAs form the basis of downstream analyses, including phylogenetic tree construction and motif identification, which are crucial for understanding genetic variation and evolutionary biology.

Manual editing of alignments is often necessary to correct errors, refine sequences, or adapt alignments for specific analyses, highlighting the importance of user-friendly tools. Although useful for some user groups, command-line interfaces (CLI) and programming languages do not offer a desirable option for most biologists. Graphical user interfaces (GUIs) are particularly advantageous for such users, as they are more intuitive and easier to navigate.

Sequence alignment viewers, editors, and analysis tools are essential for visualizing, modifying, and interpreting sequence and alignment data. Despite the availability of several sequence analysis tools, many exhibit significant limitations. Most tools implement file parsers, biological functions, and analytical methods independently, without leveraging established bioinformatics libraries. This approach complicates development, reduces maintainability, and weakens future scalability.

Moreover, available tools support only a limited range of file formats, some operate exclusively on specific operating systems, or fail to provide intuitive designs despite integrating advanced editing and analytical components. Many also lack crucial features such as annotation viewers and editors, which are especially important for

handling file formats like GenBank and EMBL that store rich metadata. While some programs offer a wide range of features, their complexity often hinders usability and reduces productivity.

Motif analysis, another key feature, is often underdeveloped. Many existing tools only offer basic pattern search functionalities, solving single-pattern matching problems but neglecting more complex tasks such as multiple-pattern matching or approximate matching using Position-Specific Scoring Matrices (PSSMs). Additionally, some tools have not been updated in years—BioEdit [1], for example, remains popular but has seen no updates since 2007—making them less suitable for modern datasets.

Finally, the rapid growth in sequence and alignment file sizes, due to advancements in sequencing technologies, poses a challenge, as not all tools are equipped to process such large datasets efficiently. These issues highlight the need for a modern, cross-platform, user-friendly tool capable of addressing these gaps.

This thesis proposes the development of a sequence analysis tool built on Biopython [2], a widely adopted and continuously evolving bioinformatics library. The primary objectives of the study are:

- To integrate fundamental features of existing tools while addressing their limitations.
- To enable seamless parsing, editing, and conversion of sequence and alignment files in numerous formats, supported by Biopython's extensive capabilities.
- To incorporate advanced functionalities such as annotation viewing and editing, manual and automatic sequence aligment, exact and approximent motif analysis, and memory-efficient handling of large files.

By leveraging Biopython, the proposed tool ensures support for a wide range of file formats, streamlines the integration of biological operations, and facilitates future enhancements. The annotation editor and viewer, along with robust motif analysis

capabilities, address gaps in existing tools. Additionally, the tool's cross-platform compatibility ensures accessibility for researchers using diverse operating systems.

This work aims to meet the growing need for a modern, adaptable, and memory-efficient sequence analysis tool, paving the way for future advancements. Yet, the initial version of the tool will not be as comprehensive as well-established software like Jalview [3], which benefits from over two decades of development. However, the platform is designed to be extensible and open to future improvements. The proposed tool incorporates key components for sequence analysis:

- Sequence: Allows users to view, edit, and manipulate sequences, including biological operations such as transcription, translation, and reverse complements.
- Alignment: Provides pairwise and multiple sequence alignment functionalities by presenting interfaces to many external tools like Clustal Omega [4] and MUSCLE [5]. It includes additional functionalities for manual modifications, as well.
- Annotation: Allows annotations to be viewed and edited in a dedicated window open for a specific sequence. Supports a variety of sequence based and letter based annotation types.
- Phylogenetic Tree: Supports parsing and visualization using FigTree [6], along with tree inference from alignments using FastTree [7] and RAxML [8]. It also gives the option for conversion of tree file formats.
- Motif: Enables the creation, editing, and conversion of motif files, along with advanced search capabilities using exact matching and PSSM-based approximate matching.

Apart from these main components, it offers file operations such as opening numerous file formats, indexing files for large datasets and conversion among various file formats. Additionaly, common edit operations such as undo-redo, copy-paste, delete-insert are also presented. Furthermore, a bunch of select and view options are given for easier interaction. All these components are explained in detail in this thesis, which is structured as follows:

- Chapter 1: Introduction – Provides background, problem statement, objectives, and the scope of the research.
- Chapter 2: Literature Review – Discusses related works, highlighting their strengths, limitations, and relevance to the proposed tool.
- Chapter 3: Methodology – Describes the design and implementation of the proposed tool, detailing its components and functionalities.
- Chapter 4: Results and Evaluation – Presents the tool's performance, including memory efficiency and functionality comparisons with existing alternatives. Addresses limitations, and discusses potential improvements.
- Chapter 5: Conclusion – Summarizes the study and outlines future development plans.

2. LITERATURE REVIEW

With the rapid advancements in bioinformatics, a large number of sequence alignment and analysis tools have been developed to address the growing needs of genetic data processing. These tools vary in their capabilities, catering to different aspects of sequence analysis. This section evaluates four popular tools—Jalview [3], Seaview [9], AliView [10], and Seqotron [11] based on their features, usability, and robustness. While commonly referred to as "alignment editors", these tools are more comprehensive in their functionality, offering features such as biological sequence operations (e.g., reverse complement, translation), phylogenetic tree creation, and advanced analyses.

These tools were chosen based on their widespread usage in the field, their ability to efficiently handle large datasets, their recent updates indicating active development and support, and their relevance to the features and functionality proposed in the developed tool. Furthermore, they are among the most adopted and well-established tools in the domain and published on prestigious journals. By analyzing their strengths and limitations, this study identifies gaps that our application aims to address.

Jalview is a widely used bioinformatics tool developed by The Barton Group at the University of Dundee. With contributions from 18 primary developers, it has been continuously developed since 1995, with the latest version released in September 2024 [12]. Funded by BBSRC and the Wellcome Trust, Jalview has gained significant recognition in the field, as evidenced by its Bioinformatics (Oxford University Press) paper, which has been cited 10,347 times. The software provides publicly available servers for sequence alignment and analysis, having processed over 442,000 jobs, making it a vital resource for researchers worldwide [12].

Seaview is a bioinformatics tool developed by researchers from Claude Bernard University Lyon 1, Montpellier 2 University in France, and the University of Auckland in New Zealand [9]. Supported by CNRS, a major government research agency, Seaview

has been actively developed since 1996, with its latest version released in 2021 [13]. Its significance in the field is reflected in its paper published in Molecular Biology and Evolution (Oxford University Press), which has been cited 2,712 times. Seaview provides essential tools for sequence alignment and phylogenetic analysis, making it a valuable resource for researchers [9].

AliView is a bioinformatics tool developed at the Department of Systematic Biology, Uppsala University. It originated during the 1000 Plants (1KP) Project, a large-scale international research initiative [10]. Funded by Formas, a Swedish government research council, AliView has been actively developed since 2014, with its latest version released in 2021 [14]. Its impact is demonstrated by its paper published in *Bioinformatics* (Oxford University Press), which has been cited 3,426 times [10]. AliView is widely used for sequence alignment visualization and editing, providing an efficient and user-friendly platform for researchers.

Seqotron is a bioinformatics tool developed by researchers from the University of Technology Sydney and the University of Sydney [11]. It was funded by the National Health and Medical Research Council (NHMRC), a government research council [11]. First introduced in 2015, Seqotron provides a user-friendly platform for sequence analysis [15]. Its impact is reflected in its publication in BMC Research Notes (BioMed Central), which has been cited 54 times [11]. The tool is designed to facilitate efficient sequence data handling and analysis for researchers in molecular biology and related fields.

The counterparts selected for this work—Jalview, Seaview, AliView, and Seqotron—are well-established bioinformatics tools, each developed by experienced researchers in the field. These tools have been continuously refined over many years, backed by academic institutions and government funding, ensuring their credibility and reliability. The careful selection of these tools ensures a fair and rigorous comparison, as each represents a well-established and highly regarded solution within the bioinformatics community.

In this study, the selected tools are compared based on a range of feature categories to provide a comprehensive analysis of their capabilities. These categories include file operations, sequence operations, alignment functionalities, phylogenetic tree construction, motif search, annotation handling, analysis features, cross-platform compatibility, and their ability to handle large files. By dividing the comparison into these categories, we aim to systematically evaluate the tools' strengths, limitations, and unique features.

2.1. File Operations

The ability to open and save files in various formats is a critical feature for sequence alignment tools, enabling compatibility with different data sources and workflows. The tools reviewed here support a range of file formats, with additional functionalities such as exporting alignments or retrieving sequences from databases.

Jalview supports an extensive range of file formats, opening 19 types, including Fasta, PFAM, Stockholm, PIR, BLC, AMSA, HTML, RNAML, JSON, PileUp, MSF, Clustal, PHYLIP, GenBank, ENA Flatfile, GFF, PDB, mmCIF, and its own Jalview format [16]. Additionally, it allows users to read alignments from a given URL, enhancing convenience when working with online resources [16]. Jalview also integrates with public databases such as UniProt, Pfam, Rfam, and the PDB, enabling direct retrieval of sequences and alignments [16]. For output, it provides options to export alignment figures in various formats, including HTML, EPS, and PNG, maintaining a WYSIWYG (What You See Is What You Get) editing interface for consistent visual representation between screen and export [16].

Seaview supports six file formats: Fasta, interleaved Phylip, Clustal, multiple sequence files of the Genetics Computer Group package, Nexus, and Mase [9]. It facilitates network-based sequence retrieval from databases like GenBank, EMBL, and UniProt/SwissProt, and allows importing feature table elements such as coding sequences and rRNA from nucleotide databases [9]. The tool also supports saving specific

sequences or sites to files, offering flexibility in selective data export [17]. Additionally, it provides options to create alignment visualizations in PDF or PostScript formats for publication-quality output [17].

AliView handles six file formats, including FASTA, FASTQ, Phylip, Nexus, Clustal, and MSF [14]. It offers specialized export options, such as saving selected sequences or regions as a Fasta file [14]. For nucleotide datasets, it supports saving alignments as Codonpos Nexus files, where nucleotides are categorized into charsets by codon positions and non-coding regions [14]. Additionally, the tool enables exporting alignments as PNG images and provides a "print to file" option to generate PostScript files for alignment visualization [14].

Seqotron supports eight file formats: FASTA, NEXUS, PHYLIP, MEGA, CLUSTAL, Stockholm, GDE, and NBRF [11]. Additionaly, it offers alignment export to PDF files.

This comparison highlights Jalview's superior format compatibility and database integration, while Seaview and AliView offer notable features for selective data export and high-quality visualization.

2.2. Sequence Operations

Sequence operations, such as translation, reverse complement, and consensus computation, are fundamental for processing and analyzing sequence alignments. The tools under review provide a range of these functionalities, catering to both basic and advanced sequence manipulation needs.

Jalview provides a cDNA translation function that uses the standard codon translation table specified in its online documentation. This feature translates nucleotide alignments or selected regions into aligned peptide sequences, facilitating downstream protein analysis [16].

Seaview supports versatile sequence operations, including nucleotide-to-protein translation using user- or database-defined genetic codes [9]. Additionally, it allows the creation of complementary and reverse sequences, enabling strand-specific analyses [17]. The tool includes a "Consensus sequence" function to compute the consensus of selected sequences, and users can exchange Us and Ts in selected sequences, providing flexibility in working with RNA and DNA data [17].

AliView includes options to translate nucleotide sequences into amino acid sequences [14]. It supports reverse complement, reverse, and complement transformations for individual sequences or entire alignments, making it suitable for tasks requiring sequence orientation adjustments [14]. AliView also allows merging two selected sequences; in cases of non-identical overlaps, it generates a consensus sequence [14].

Seqotron offers functionality to concatenate sequences with identical names, streamlining the process of assembling sequences from different sources [11]. It supports nucleotide-to-protein translation using one of the 18 genetic codes available, offering flexibility for diverse genomic datasets [18]. Additionally, the tool calculates complement and reverse complement sequences, further expanding its capabilities for nucleotide sequence manipulation [18].

In summary, Jalview offers codon translation, while others provide a broader range of sequence operations, including sequence merging, taking (reverse) complements and DNA-RNA transition.

2.3. Alignment

Alignment functionality is central to sequence analysis, as it enables the comparison of genetic sequences for evolutionary, structural, and functional studies. The tools reviewed provide a variety of alignment options, from pairwise alignments to profile-based methods and integration with external alignment programs.

Jalview supports optimal pairwise alignments between any sequences [16]. It integrates multiple alignment algorithms, such as ClustalW [19], Muscle [5], MAFFT [20], ProbCons [21], T-COFFEE [22], and Clustal Omega [4], offering flexibility for diverse user needs [16]. Users can access Jalview's dedicated SOAP web services for alignment tasks, and predefined presets optimize alignments based on speed or accuracy [3,16]. Jalview also allows profile alignment, though this is currently limited to Clustal Omega and ClustalW [16]. Moreover, the tool enables parameter customization for web services, including substitution score matrices, gap penalties, and distance metrics used for guide tree construction [16].

Seaview allows users to align sequences using algorithms such as ClustalW and Muscle, with the option to apply alignment to all or only selected sequences [9,17]. The tool supports profile alignment, where new sequences can be aligned against pre-aligned groups, using Muscle or ClustalW [17]. Additionally, Seaview provides the flexibility to drive any external alignment program capable of processing FASTA-formatted data, making it adaptable for custom workflows [9].

AliView supports invoking external alignment programs, with Muscle included by default and others requiring installation [10]. It offers realignment options, including the ability to realign a selected block of sequences or translate nucleotide sequences into amino acids for alignment, followed by reverting to nucleotide sequences [14]. AliView also supports profile alignment and provides an external interface for seamless integration with additional programs [14].

Seqotron facilitates alignment using built-in MUSCLE and MAFFT algorithms [11]. It includes a "Transalign" feature that aligns protein-coding DNA sequences by translating them into amino acids during the alignment process [18]. Additionally, Seqotron can refine existing alignments using the MUSCLE algorithm with the '-refine' option, enabling iterative improvement of alignments [18].

In summary, Jalview stands out with its extensive algorithmic integration, parameter customization, and unique web server support for alignment tasks, making it particularly well-suited for long-running alignments. In contrast, Seaview and AliView focus on external program support and profile alignment capabilities.

2.4. Annotations

Annotation functionality varies significantly across sequence alignment tools, with some providing extensive support for importing and visualizing annotations, while others lack these capabilities.

Jalview supports relatively more annotations, particularly for alignment-related and sequence-specific annotations. It calculates alignment-related annotations such as conservation, quality, and consensus scores [16]. Sequence-specific feature annotations, such as domains and binding sites, can be linked to specific residues and imported from public databases or file formats like GFF, Stockholm, AMSA, and Jalview-specific formats [16]. However, it can not retrieve annotations from files such as GenBank, EMBL and UniProt although they are quite common in the field. Also, it can not save annotations in GenBank, EMBL or UniProt formats, which appears as an important limitation.

Seaview, in contrast, offers more limited annotation support. It can handle basic, unstructured annotations such as "comments" and alignment-based annotations like "footers", but this functionality is restricted to specific file formats, including Mase and Nexus [17]. While useful for basic annotation needs, Seaview's annotation capabilities are not as comprehensive as Jalview's.

AliView and Seqotron lack support for annotations altogether. These tools focus exclusively on handling sequence names and sequences, without the ability to import or manage annotation data [14, 18]. As a result, users requiring annotation support must rely on external tools or additional workflows.

In summary, Jalview stands out for its better annotation support, offering sequence-specific and alignment-related annotations. Seaview provides limited annotation capabilities, while AliView and Seqotron do not support annotations, reflecting their focus on core alignment features.

2.5. Phylogenetic Tree Construction and Visualization

Phylogenetic tree construction and visualization are critical for understanding evolutionary relationships among sequences. Each tool provides distinct functionalities for tree building, visualization, and manipulation, tailored to specific research needs.

Jalview calculates phylogenetic trees from distance matrices, using either percentage identity or aggregate BLOSUM62 scores. These trees can be built with Average Distance (UPGMA) or Neighbor Joining algorithms [16]. Users can import external trees and export constructed trees in Newick format [16]. Jalview features an integrated tree viewer, which allows interactive visualization; clicking on the tree enables the automatic partitioning and coloring of sequences, aiding in intuitive data interpretation [16].

Seaview integrates PhyML version 3 [23] for maximum-likelihood tree construction, leveraging its capabilities as an independent program [9]. Tree building can be applied to all sequences or a subset of selected sequences and sites [9]. Most PhyML options can be configured through Seaview's graphical interface, ensuring flexibility for advanced users [9]. The built-in tree viewer supports various customization options, with trees exportable in Newick, PDF, or PostScript formats [9]. Additionally, trees can be printed or copied to the clipboard, facilitating integration with external applications such as office softwares [9].

AliView provides a streamlined process for tree generation. It includes a preconfigured button that directs alignments to FastTree [7] for rapid phylogenetic tree construction.

Once calculated, the tree is automatically opened in FigTree [6] for further visualization and analysis [10].

Seqotron offers extensive tree-building capabilities, using Physher [24] to construct trees from nucleotide or amino acid alignments [18]. It supports Neighbor Joining and UPGMA algorithms for distance-based methods and also allows maximum-likelihood tree construction [18]. Seqotron implements resampling techniques, including bootstrapping (resampling sites with replacement) and jackknifing (resampling sites without replacement), to assess branch support. These resampling processes can be parallelized by configuring the number of threads [18]. The built-in tree viewer is robust, offering functionalities such as taxa coloring, taxon name search, re-rooting, node rotation, and exporting trees to Newick-based text or PDF files [11]. Seqotron also facilitates sequence extraction based on evolutionary relationships by allowing sequence selection directly within the tree viewer [11].

In summary, Seqotron and Jalview offer versatile tree-building methods with interactive visualization, while Seaview emphasizes maximum-likelihood trees via PhyML with a user-friendly graphical interface. AliView simplifies the process by integrating FastTree with FigTree, catering to users seeking rapid tree construction and analysis.

2.6. Motif Search

Motif search functionality enables users to identify specific patterns or motifs within sequences, which is crucial for various analyses, such as identifying conserved regions or functional sites. The tools vary significantly in their motif search capabilities.

Jalview provides an relatively more detailed pattern search feature, supporting regular expressions (regex) for precise and flexible searches. This functionality allows users to search within both annotations and sequences [16].

Seaview offers a simpler search option, which identifies patterns while accounting for gaps in the alignment [17]. This feature, while less advanced than Jalview's regex-based search, is still effective for basic motif discovery across alignments.

AliView enhances motif search by supporting searches with IUPAC nucleotide codes, enabling the identification of ambiguous patterns. Additionally, it incorporates the ability to identify patterns across gaps [10].

Seqotron, in contrast, includes only a basic search feature with limited functionality, focusing on straightforward pattern identification without support for more complex criteria [18].

In summary, Jalview excels with regex-based motif searches, while AliView stands out for supporting ambiguous pattern searches using IUPAC codes. Seaview provides basic gap-aware pattern searching, while Seqotron offers only minimal motif search functionality.

2.7. Analysis and Specialized Features

Beyond alignment and tree-building functionalities, bioinformatics tools often include advanced analytical features that aid in deriving meaningful insights from sequence data.

Jalview provides a robust set of analytical tools. Principal Components Analysis (PCA) creates a spatial representation of similarities, either for a selected region or the entire alignment [16]. Protein secondary structure predictions are accomplished through integration with the Jpred [25], alongside predictions of protein disorder using multiple predictors [16]. Jalview also supports working with 3D structures by linking aligned sequences with structural data from the PDB and mapping sequences accurately using the EMBL-EBI's SIFTS database [16].

Additionally, Jalview allows for protein alignment conservation analysis by computing over 17 amino acid conservation measures, offering detailed insights into alignment properties [16].

Seaview offers a dot-plot analysis which compares two sequences to identify regions of high similarity that may have been overlooked by alignment algorithms. This visual approach aids in validating alignment accuracy and identifying conserved domains.

AliView includes features for primer design, enabling users to identify degenerate primers in conserved regions of mixed-species alignments [14]. Parameters such as primer length, degeneracy, self-binding values, and melting temperature (TM) can be adjusted in the opened window, which displays potential primers in an ordered list [14]. AliView also supports Nexus file specifications, allowing users to read, preserve, and modify Codonpos, Charset, and Excludes annotations [14].

Seqotron focuses on estimating distance matrices from nucleotide or amino acid alignments using Physher. Supported distance methods include Raw, Jukes-Cantor (JC69), and Kimura 2-parameter (K2P) for nucleotide alignments, as well as raw distance for amino acid alignments [18]. These tools provide the foundational metrics needed for downstream evolutionary and phylogenetic analyses.

In summary, Jalview stands out for its integration with 3D structural data, support for PCA and conservation analysis, while Seaview emphasizes dot-plot comparisons. AliView focuses on primer design and support for Nexus annotations, whereas Seqotron provides distance matrix estimation methods.

2.8. Handling Large Files

The ability to handle large files is critical for bioinformatics tools, as researchers can work with datasets exceeding several gigabytes. According to our tests, all four

tools demonstrated the capability to handle files up to 1 GB, as evaluated in Section 4.2. However, their memory consumption varied significantly when managing these datasets.

Among these, AliView stands out with its unique ability to handle files of virtually unlimited size. This is achieved through an efficient indexing mechanism, where sequences are indexed during file loading and only cached in memory when accessed. This approach minimizes memory usage, enabling processing of extremely large files [14]. In contrast, while Jalview, Seaview, and Seqotron are capable of loading large files, they rely on conventional methods that fully load data into memory.

Overall, AliView's indexing-based file handling provides a significant advantage for researchers dealing with massive datasets, distinguishing it from the other tools in this category.

2.9. Cross-Platform Compatibility

Cross-platform compatibility ensures that tools can be utilized across different operating systems, enhancing accessibility and usability for a diverse user base.

Seqotron is limited to macOS and does not offer support for other operating systems, restricting its user base to those with access to Apple devices [18].

In contrast, Jalview, Seaview, and AliView are fully cross-platform, providing support for major operating systems such as Windows, macOS, and Linux [10, 16, 17].

2.10. Critical Analysis and Identified Gaps

While the reviewed tools provide a robust foundation for sequence alignment and analysis, several limitations and gaps remain, which highlight opportunities for improvement.

The existing tools support a limited range of file formats, which restricts their versatility. Among them, Jalview stands out as the most flexible, supporting a wide array of formats. However, even Jalview does not accommodate all commonly used file types, leaving room for further improvement in format compatibility.

Annotations play a critical role in managing metadata-rich file formats. Tools like AliView and Seqotron lack annotation viewers and editors entirely, severely limiting their utility for formats that store detailed metadata. In contrast, Jalview offers more annotation functionality, including alignment-related and feature annotations. However, Jalview can not open annotation rich formats such as GenBank and EMBL, does not support letter-based annotations, such as PHRED quality scores, and does not save in GenBank or EMBL format.

Although Jalview provides an extensive set of features, its complex user interface hampers usability and productivity. New users often face a steep learning curve, as navigating Jalview typically requires consulting its manual. Conversely, AliView and Seqotron were found to be the most intuitive and user-friendly applications during our evaluation. These tools served as an inspiration for the user interface and user experience (UI/UX) design of our tool, emphasizing simplicity and accessibility without compromising functionality.

Another limitation is Seqotron's exclusive compatibility with macOS, making it unavailable to users on other operating systems. In contrast, the cross-platform support of an application enhances its accessibility and usability across diverse computing environments.

Motif analysis remains an underdeveloped area in all the existing tools. While all tools provide basic pattern search functionalities, their capabilities are limited to solving single-pattern matching problems. This highlights the need for more sophisticated motif analysis features to address the growing complexity of sequence data analysis in modern research.

With the exponential growth of sequence data, handling large files has become a critical requirement. Only AliView implements an efficient indexing mechanism, allowing it to open files that exceed available RAM. This capability is becoming increasingly important for researchers working with ever-growing datasets, making this functionality indispensible.

The reviewed tools provide valuable functionalities, but critical gaps remain in file format compatibility, annotation handling, motif analysis, and scalability for large datasets. Our tool seeks to address these gaps by adopting the strongest and most fundamental features of the existing tools while introducing solutions to improve file format compatibility, support for detailed annotations, advanced motif analysis, and scalability.

3. METHODOLOGY

3.1. Development Framework and Tools

The development of this tool was built using Biopython, which handles sequence parsing, biological operations, and motif search. Tkinter was chosen as the cross-platform GUI framework for building the user interface, ensuring compatibility across different systems. Additionally, two small packages were integrated to monitor memory usage. The development process spanned one year of active work, resulting in a codebase of 8,190 lines of Python code. The application is open-source, and its code along with the installation guide can be accessed online [26].

3.1.1. Biopython

Biopython [2] was chosen as the core library for handling biological data in the application. At the time of starting the development, the latest version of Biopython was 1.83, which was utilized throughout the project. The library has since been updated to version 1.85, demonstrating its ongoing development and support. Biopython was particularly beneficial in parsing numerous file formats and extracting annotations embedded within them. Additionally, it was employed for file format conversion tasks, facilitating seamless interoperability between various bioinformatics tools. Beyond parsing and conversions, Biopython supports essential sequence analysis functionalities, including transcription, translation, and complement calculation, which were leveraged frequently in the application. A significant use case was motif analysis, where Biopython was employed for opening and saving pattern files, along with searching patterns in sequences.

Biopython is a comprehensive collection of Python libraries designed for bioinformatics applications. First released in 2000, it is one of the oldest and most actively maintained bioinformatics libraries. With contributions from 18 core developers and

over 350 open-source contributors, Biopython has established itself as a reliable and widely used tool in the field. Its significance is highlighted by its publication in Bioinformatics (Oxford University Press), which has been cited 5,517 times, reflecting its strong impact on the bioinformatics community.

Biopython was selected for several compelling reasons. Being Python-based, it is easy to use and integrates seamlessly with the rest of the application. Its status as an established, well-documented library with a long history of development, a large user base, and active contributors ensures reliability. As an open-source project, Biopython is not only free to use but also allows for customization to meet specific project requirements. The active developer community ensures regular updates and timely bug fixes, further enhancing its appeal for bioinformatics applications.

Furthermore, Biopython offers significant advantages for bioinformatics software development. It accelerates the development process by providing efficient parsers for various biological file formats, reducing the need for custom implementations. Additionally, it includes a comprehensive set of tools for sequence analysis, motif searching, and structural biology, making it an appropriate choice for developing a sequence analysis tool.

Alternative libraries such as BioJava [27] and BioPerl [28] were not preferred for this project. One primary reason was the lack of expertise in Perl programming, making BioPerl unsuitable. Similarly, Java was considered less favorable due to its verbosity and complexity, which are not ideal for rapid development workflows. Biopython's large and active community, which provides abundant tutorials, examples, and solutions, further reinforced its selection over alternatives that lack similar community support.

3.1.2. Tkinter

Tkinter was chosen as the graphical user interface (GUI) framework for the application. Its inclusion in Python's standard library eliminates the need for additional

installation, streamlining the setup process for developers. Tkinter also ensures cross-platform compatibility, allowing the application to run seamlessly on major operating systems, including Windows, macOS, and Linux.

Tkinter is known for its simplicity and ease of use, enabling developers to quickly design and implement GUIs with minimal effort. Unlike more complex frameworks such as PyQt [29], which offers advanced features but comes with a steeper learning curve and additional installation requirements, Tkinter requires minimal boilerplate code and is built into Python, making it ideal for rapid development. Its support for scalable widgets, particularly the *Text widget*, is crucial for displaying and editing large text-based datasets such as biological sequences, making it a practical and accessible choice for developing the application's user interface.

3.1.3. Additional Tools and Libraries

In addition to Biopython and Tkinter, several other libraries and tols were employed to enhance the functionality and performance of the application.

Memory Profiler [30] was utilized to monitor the application's memory usage during execution. This tool allowed for the identification of memory bottlenecks and ensured that the tool remained efficient, even when handling large biological datasets.

Pympler [31] was used for tracking and analyzing Python objects. It provided valuable insights into the memory consumption of different objects during runtime, helping optimize memory usage significantly.

PyInstaller [32] was chosen for distributing the application as a standalone executable. PyInstaller converts Python applications into self-contained executables, making it easier to deploy the tool without requiring users to install Python or additional dependencies. This helped speed up the distribution process and ensure compatibility across different platforms.

3.2. Platform Overview

The graphical user interface (GUI) of the tool is designed with modularity in mind, ensuring that each menu contains only related features for ease of navigation and usability. For the explanation of file formats mentioned in this section, please refer to the Biopython documentation [33, 34].The list for menu names is as follows:

- File: Includes options for opening, saving, and converting files.
- Edit: Provides tools for editing the sequences within the application.
- Select: Allows users to select specific regions or sequences for further analysis or editing.
- View: Includes display settings, such as zoom or color adjustments.
- Sequence: Contains functions for sequence operations, including transcriptions, translations, and complementing sequences.
- Alignment: Offers features for sequence alignment tools and manual alignment.
- Annotation: Provides tools for managing sequence annotations, including adding, editing, or deleting annotations.
- Tree: Contains options for viewing and inferring phylogenetic trees using external tools, as well as tree file conversion.
- Motif: Offers features related to motif analysis, including motif searching and working with motif files.
- Other Tools: Provides easy access to custom tools and programs added by user.

3.2.1. File

All file operations are performed using Python threads to ensure that the user interface does not freeze during the process. A popup window is displayed to inform the user of the ongoing operation.

3.2.1.1. Open Sequence File. This feature allows users to open a sequence file. The application sets the file type as “Sequence”, indicating that the file contains unaligned

sequences rather than an alignment file. Each sequence in the file is treated independently and can have arbitrary types (i.e. DNA, RNA, Protein) and lengths.

The tool supports a wide range of file formats, with a total of 34 options: abi, abi-trim, ace, fasta, fasta-2line, ig, embl, embl-cds, gb, gck, genbank, genbank-cds, imgt, nib, cif-seqres, cif-atom, pdb-atom, pdb-seqres, phd, pir, fastq, fastq-sanger, fastq-solexa, fastq-illumina, qual, seqxml, sff, snapgene, sff-trim, swiss, tab, twobit, uniprot-xml, and xdna.

Additionally, the application supports compressed formats in `gz` and `bz2`. Sequence file dialog window is demostrated in Figure 3.1.

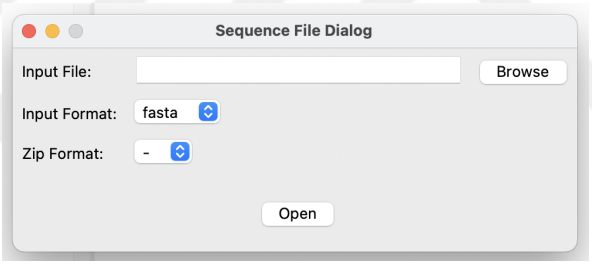


Figure 3.1. Sequence file dialog window.

When opened, the file is displayed in the application as shown in Figure 3.2. The left panel displays the sequence names or IDs, while the right panel is dedicated to displaying the sequence content.The first row of the right panel contains a ruler, displaying sequence positions with markers at every 10th position to indicate length. The file type, either “Sequence” or “Alignment”, is indicated on the left side of the bottom panel, “Sequence” in this case. Certain editing operations are enabled only for “Sequence” or “Alignment” files, while others are common. The row and column values, which show the mouse position in real time, are displayed on the right side of the bottom panel.

3.2.1.2. Open Sequence File as Index. Index mode allows the user to open large datasets that exceed RAM limitations, which is missing in most alternatives except AliView.

Figure 3.2. Sequence viewer window.

However, editing capabilities are restricted in this mode. The supported formats include 19 file types: ace, embl, fasta, fastq, fastq-sanger, fastq-solexa, fastq-illumina, genbank, gb, ig, imgt, phd, pir, sff, sff-trim, swiss, tab, qual, and uniprot-xml.

In this mode, the file is initially indexed completely, and then the sequences in the visible area are dynamically read into memory and shown as the user scrolls up or down. For user's point of view, the file is displayed in the application in the same way as regular opening, as shown in Figure 3.2.

Indexing the opened file is performed by Biopython by marking the start offsets of each sequence. Since Tkinter does not natively support this functionality, we implementated the rendering part where the visible region is calculated as the user navigates and only those sequences are inserted into the Text widget. This approach creates the illusion that the entire content is loaded, offering seamless navigation for users. However, due to the nature of this mode, a limited list of file formats is supported.

To enhance efficiency, we optimized small changes like slightly scrolling up/down or resizing the window by calculating the difference between the previous and next

states. Instead of redrawing the entire screen, we incorporate newly appearing sequences while removing disappearing ones to improve performance. However, due to the nature of this mode, a limited set of file formats is supported.

The rendering speed in index mode largely depends on the disk read speed, as sequences are fetched in real-time. For users who wish to work on specific smaller regions of very large files, the application provides an option to specify the "Number of sequences to be cached in index mode". The most recently viewed sequences, up to this specified number, are cached in memory to facilitate faster navigation within this subset of the file.

3.2.1.3. Open Alignment File. Allows the user to open an alignment file and sets the file type to "Alignment." This option requires sequences to have a fixed size, as they constitute an alignment. The alignment sequences are shown in the same way with sequence files, like the Figure 3.2. No zip formats are supported in this case. The supported formats are: (13) clustal, emboss, fasta, fasta-m10, ig, maf, mauve, msf, nexus, phylip, phylip-sequential, phylip-relaxed, and stockholm.

Typically, a file contains a single alignment. However, in cases where the file contains multiple alignments, a selector is provided to allow the user to choose an alignment based on its order in the file, as shown in Figure 3.3. Additionally, the user can switch between alignments using the "Switch Alignment" button in the "Alignment" menu.

3.2.1.4. Save File. This option allows the user to save the currently worked file. During the save process, the user is prompted to select which annotations to keep, as shown in Figure 3.4. Annotations that are not supported by the selected format are omitted. The supported formats for sequence files are: (18) fasta, fasta-2line, gb, genbank, embl, imgt, nib, phd, pir, fastq, fastq-sanger, fastq-solexa, fastq-illumina, qual, seqxml, sff, tab, and xdna.

For alignment files, the supported formats are: (8) clustal, maf, mauve, nexus, phylip, phylip-sequential, phylip-relaxed, and stockholm.

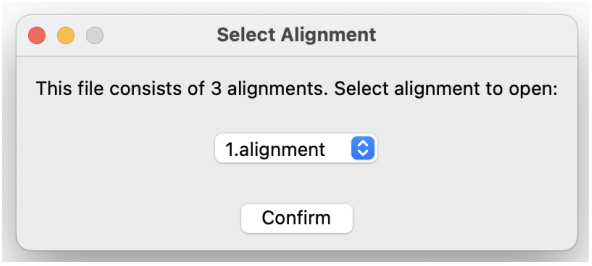


Figure 3.3. Alignment selector.

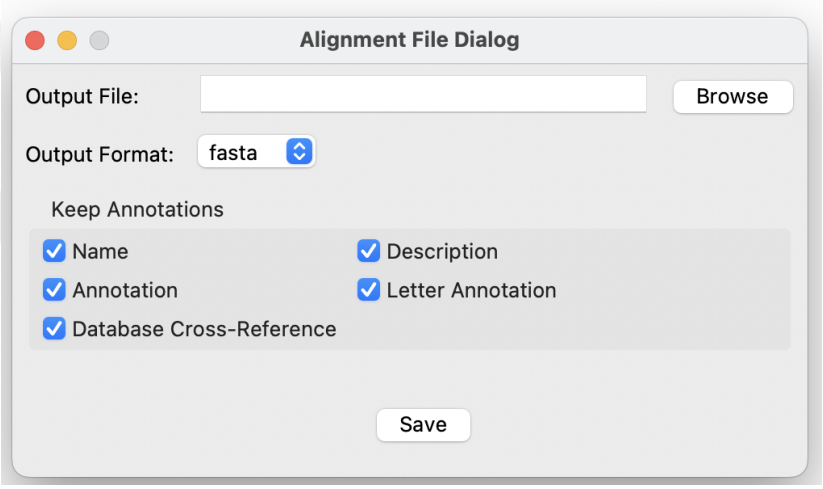


Figure 3.4. Save window.

3.2.1.5. Convert Sequence File Format. This option allows the user to convert the format of a sequence file without the need to open it. This process is designed to be memory efficient, as records are read and written one by one, without loading the entire file into memory at once. This ensures that large files can be converted without overwhelming system resources.

The input formats for this option are the same as those in the “Open Sequence File” option. The output formats available for conversion are the same as those supported in the “Save File” option for sequence files.

3.2.1.6. Convert Alignment File Format. This option allows the user to convert an alignment file from one format to another without opening it. This process is performed in a memory-efficient manner, similar to “Convert Sequence File Format”. The input formats for this option are the same as those in the ”Open Alignment File” option. The output formats available for conversion are the same as those supported in the ”Save File” option for alignment files.

3.2.1.7. Settings. This menu item opens a configuration interface with several tabs: General, Alignment Tools, Tree Tools, and Other Tools.

- General: This tab provides general application settings, as shown in Figure 3.5. The available options are explained throughout the document as related sections appear.

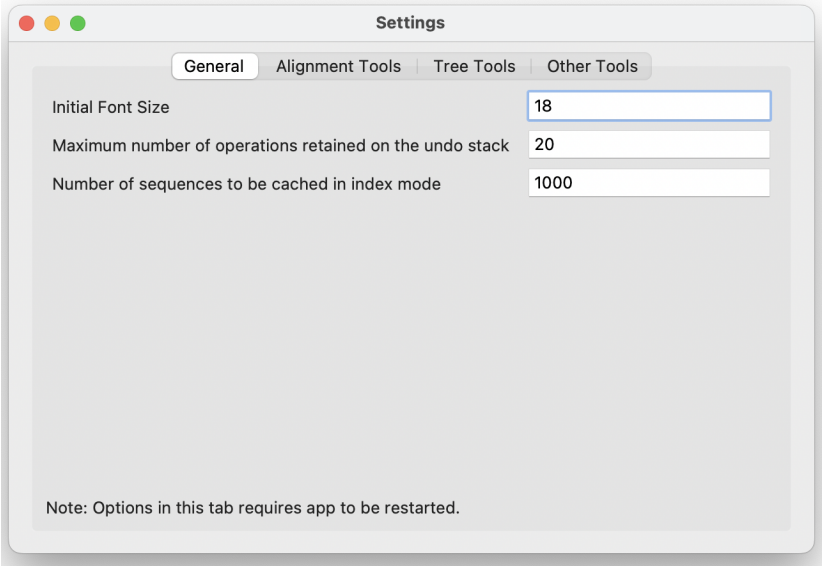


Figure 3.5. General settings.

- Alignment Tools: In this tab, the user can configure external alignment tool parameters, as demonstrated in Figure 3.6. Currently, the application supports interfaces to these tools: Muscle, ClustalOmega, Prank [35], Mafft, Probcons, MSAProbs [36] for multiple sequence alignment. Also, Needleman-Wunsch and Smith-Waterman from EMBOSS package [37] for pairwise alignment. For alignment tools, users can specify input, output and output file format for MSA, as well as gap open and gap extension parameters for pairwise alignments. These tools can be initiated later via the Alignment menu.

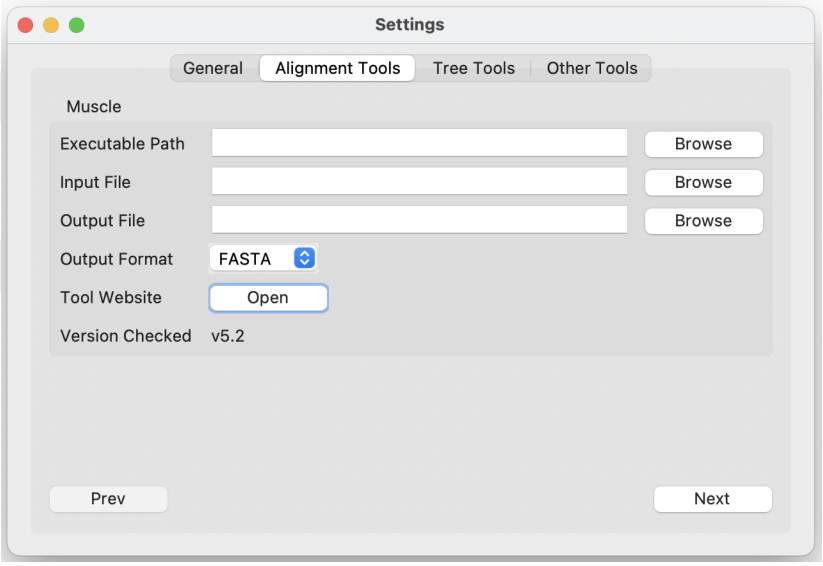


Figure 3.6. Alignment tools settings.

- Tree Tools: This tab allows users to configure external tree tool parameters, as shown in Figure 3.7. The application currently supports RAxML and FastTree for phylogenetic tree inference, with FigTree for visualization. Users can specify input and output file parameters, along with some additional tool-specific options. These tools can be run later via the Tree menu.
- Other Tools: In this tab, users can add custom external tools by entering their respective commands, as shown in Figure 3.8.. This provides immediate access to the user's preferred bioinformatics programs, which can be executed later via the "Other Tools" menu.

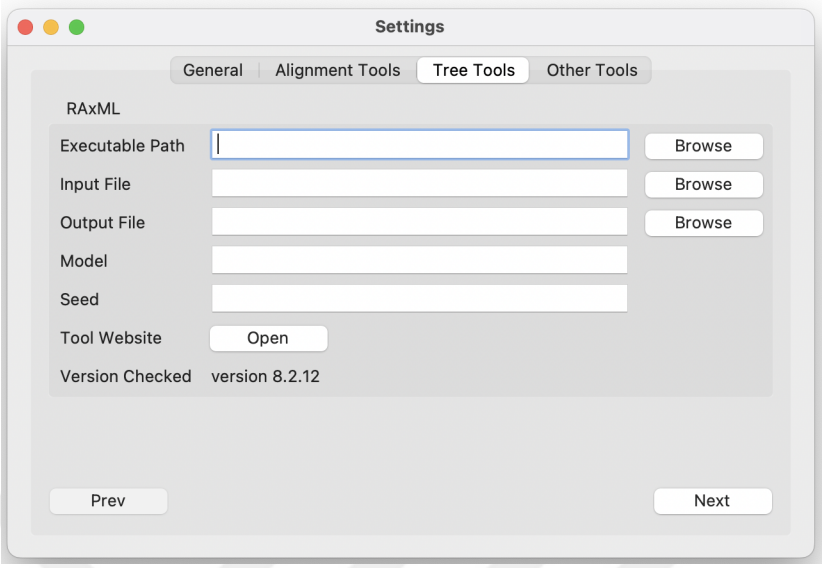


Figure 3.7. Tree tools settings.

3.2.2. Edit

3.2.2.1. Editor Mode (Keyboard). The application allows users to toggle editing mode. When editing mode is enabled, users can directly replace letters in the sequence using keystrokes. This includes the ability to insert characters from the DNA, RNA, or protein alphabets, as well as the indel-gap character (-).

3.2.2.2. Undo-Redo. The application supports undo and redo functionality, similar to a typical text editor. Users can revert changes or reapply them as needed, providing flexibility during sequence editing. The "Maximum number of operations retained on the undo stack" can be configured in the settings menu. Increasing this limit allows the application to track a larger history of editing operations while requiring additional memory. By default, no limit is imposed on the undo stack. The "Save Current State" option allows users to save the current state of the editor, preventing the user from reverting changes made before this saved state. This feature is particularly useful for locking in progress during extensive editing sessions.

3.2.2.3. Copy-Paste. The application provides the following options for copying and pasting sequence data:

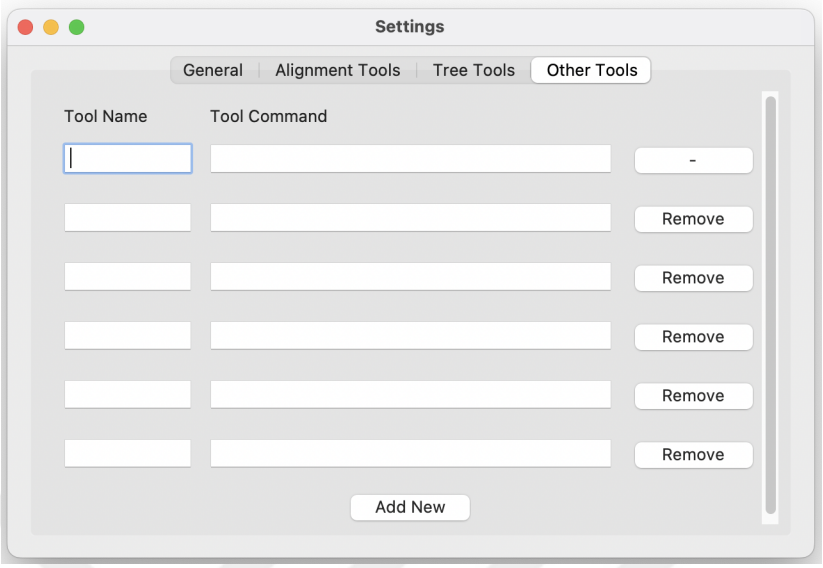


Figure 3.8. Other tools settings.

- Copy Selection as FASTA: Copies the selected sequences or regions in FASTA format, including sequence headers.
- Copy Selection as Text: Copies only the letters from the selected sequences or regions in plain text format, excluding headers.
- Paste FASTA: Inserts sequences from the clipboard (in FASTA format) directly into the application. This feature allows seamless integration of external sequence data.

3.2.2.4. Delete-Insert. The application provides the following options for deleting and inserting sequence data:

- Delete Selected: Deletes the selected sequences entirely.
- Delete Selected Letters: Deletes only the selected letters within sequences.
- Insert Unknown Letter: Inserts an unknown letter (?) at the selected position in the sequence.

3.2.3. Select

Selection can be made using the mouse across three distinct panels: the left panel (sequence names), the right panel (sequence content), and the top panel (ruler).

- Left Panel (Sequence Names): Dragging across sequence names selects entire sequences, including all their letters.
- Right Panel (Sequence Content): Dragging across this panel selects only a specific segment of the sequences, allowing for precise control over the selection range.
- Top Panel (Ruler): Dragging across the ruler selects entire columns (sites), enabling the selection of specific alignment positions across all sequences.

An example selection is shown in Figure 3.9. The following options are also offered for selection, drawing inspiration from AliView's intuitive implementation of these features:

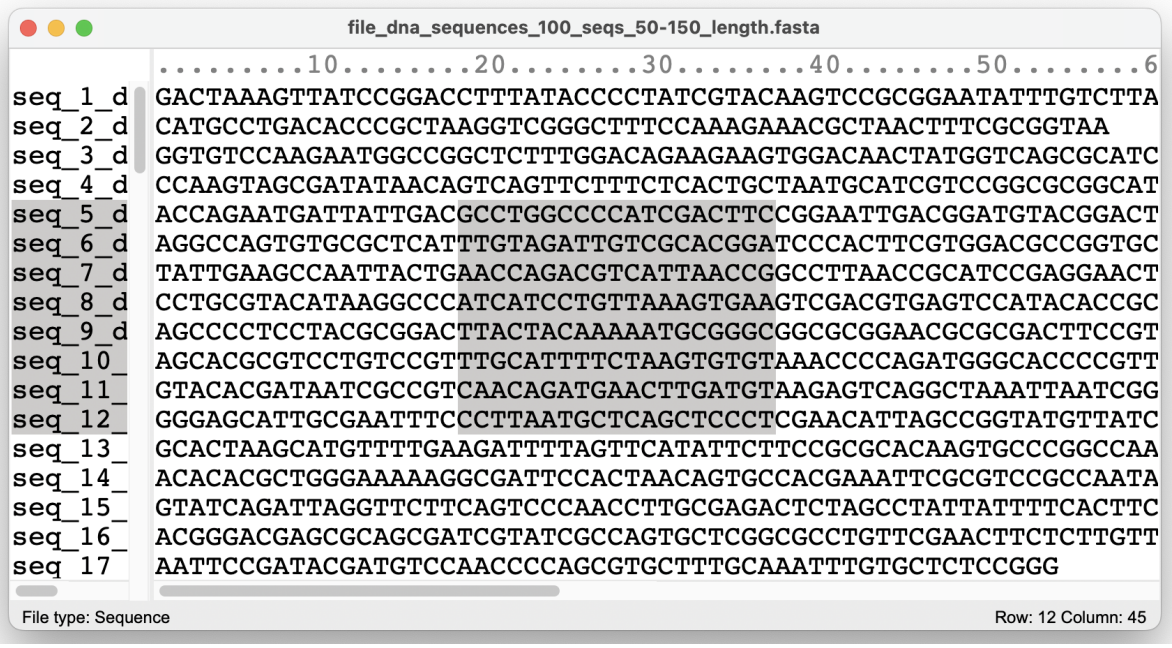


Figure 3.9. Example selection.

- De-select: Removes the current selection.
- Select All: Selects all sequences.
- Expand Selection Right: Expands the selection to the right.
- Expand Selection Left: Expands the selection to the left.
- Expand Selection Up: Expands the selection upward.
- Expand Selection Down: Expands the selection downward.
- Move Selected Sequences Up: Moves the selected sequences upward.
- Move Selected Sequences Down: Moves the selected sequences downward.
- Move Selected Sequences Top: Moves the selected sequences to the top.
- Move Selected Sequences Bottom: Moves the selected sequences to the bottom.

3.2.4. View

- Color Mode: Switch between color modes: no color, foreground, and background. Coloring is performed based on the letter, with each letter having a unique color, as shown in Figure 3.10.
- Font Size: Allows adjustment of font size in the range of 8-32, using the Courier font. Render speed is affected by this setting; as the font size decreases, more letters become visible, making rendering more challenging.
- Navigate to Position: Allows the user to go to an arbitrary position, specified by the line and column number.

3.2.5. Sequence

The Sequence component provides tools for managing and manipulating nucleotide and protein sequences. Users can add new sequences, perform transformations like concatenation, complementation, reverse complementation, transcription, and translation, and configure relevant options for each operation.

3.2.5.1. Add New Sequence. This feature allows users to add new sequences to the application in two ways. A single sequence can be added by manually entering its

sequence ID and content. Alternatively, users can add multiple sequences by opening a file and appending its contents to the existing sequences. This ensures flexibility in handling new data, whether entered manually or imported from external files.

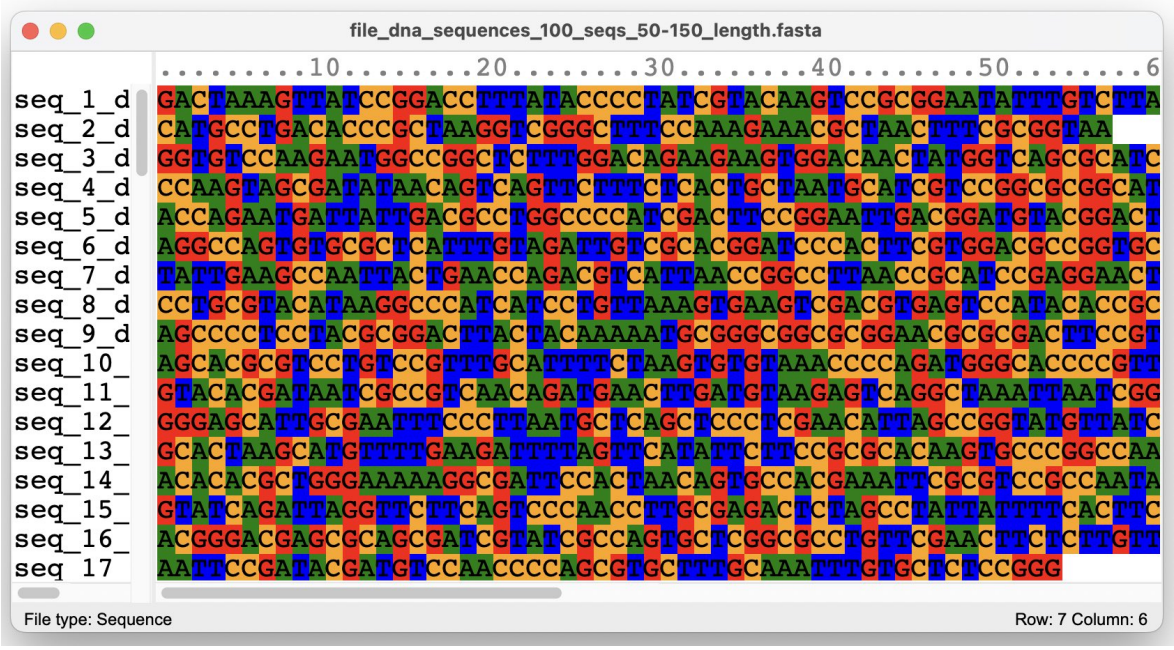


Figure 3.10. Background color mode.

For the following operations, although annotations are unlikely to apply to the newly generated sequence, we strive to retain as many annotations as possible from the original sequence (if they exist) for cautiousness. The user has the option to delete or edit these annotations later if necessary. The necessary transformations for annotations are applied where applicable; for example, letter annotations are reversed if a reverse complement operation is performed.

3.2.5.2. Concatenate Selected. This operation allows the user to concatenate the selected sequences into a single sequence.

3.2.5.3. Complement Selected. This feature provides the ability to compute the complement of selected sequences. It supports both DNA and RNA, replacing each nucleotide with its complement.

3.2.5.4. Reverse Complement Selected:. This operation generates the reverse complement of the selected nucleotide sequences. It supports both DNA and RNA, reversing the sequence and replacing each nucleotide with its complement.

3.2.5.5. Transcribe Selected. Transcription options are available for converting nucleotide sequences into RNA. Users can choose to transcribe sequences from either the coding strand or the template strand. The resulting RNA sequences are generated based on the selected strand.

3.2.5.6. Translate Selected. This feature allows the translation of selected nucleotide sequences into protein sequences. The user can configure the following options as shown in Figure 3.11:

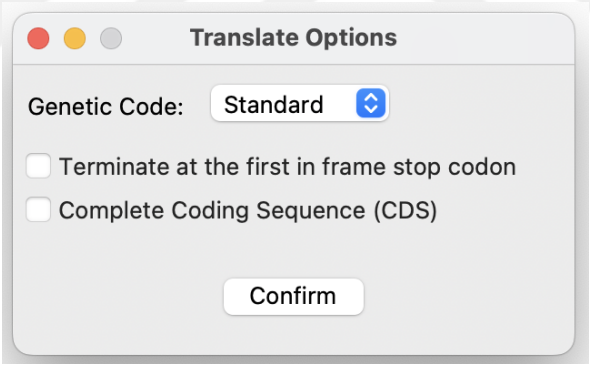


Figure 3.11. Translate window.

- Genetic Code Table: The translation table to use can be selected. The available tables are based on the NCBI's genetic code tables, as supported by Biopython.
- Stop at First In-Frame Stop Codon: This option enables translation to stop at the first in-frame stop codon, mimicking natural biological processes.
- Complete Coding Sequence (CDS): If the nucleotide sequence is a complete coding sequence (CDS), the user can indicate this by selecting the corresponding option. This ensures the sequence is validated as a proper CDS, and an error will be raised if it is not.

3.2.6. Alignment

The Alignment component provides robust support for both automated and manual sequence alignment. It includes interfaces for running popular external tools like Muscle, ClustalOmega, and Mafft for multiple sequence alignments, as well as EMBOSS tools for pairwise alignments. Users can monitor alignment progress in real-time and configure tool parameters directly within the app. Additionally, the Manual Alignment feature allows for precise adjustments and corrections to ensure high-quality alignments, addressing errors that automated methods may introduce. This comprehensive approach makes the alignment process flexible and user-friendly.

3.2.6.1. Switch Alignment. Switch among alignments in a single file if the file contains multiple MSAs (e.g., Stockholm format).

3.2.6.2. Multiple Sequence Alignment. The application provides intuitive user interfaces for external MSA tools such as Muscle, ClustalOmega, Prank, Mafft, Probcons, and MSAProbs. As mentioned in the settings section, users can configure these tools' parameters directly within the application. Note that the tools themselves need to be installed separately by the user. To assist with tool installation, a button has been added that allows users to easily access the official website of each tool, along with the version used for creating the corresponding command. The window for the Muscle tool is shown in Figure 3.12, as an example.

The progress of the running tool is displayed in real time within the application. This is achieved by capturing the standard output (stdout) and standard error (stderr) streams of the external tool using pipes, which are then read by separate threads. This allows the user to monitor the progress of the alignment or other tool executions as they proceed. The real-time progress display is shown in Figure 3.13.

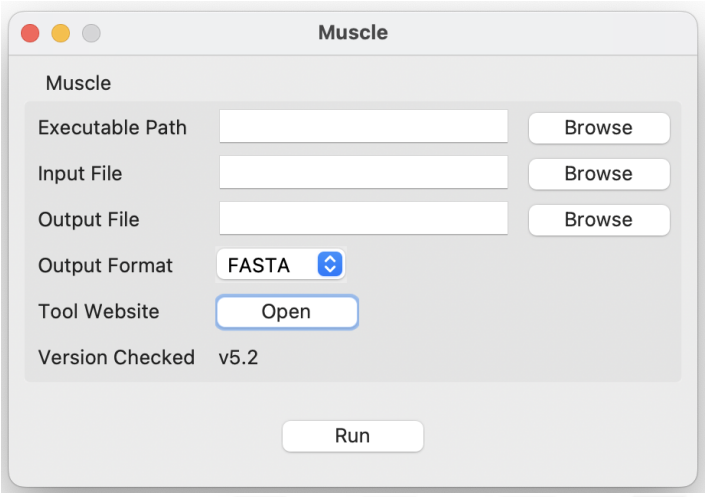


Figure 3.12. Muscle tool window.

3.2.6.3. Pairwise Alignment. The application provides intuitive user interfaces for pairwise alignment tools from EMBOSS: Needleman-Wunsch (Global) and Smith-Waterman (Local). These interfaces are designed similarly to those for Multiple Sequence Alignment (MSA) tools, as mentioned earlier. In addition to the basic alignment parameters, the user can configure gap open and gap extension values to fine-tune the alignment process. As with the MSA tools, a progress window is displayed during execution, and users can access a link to the tool's website for more information.

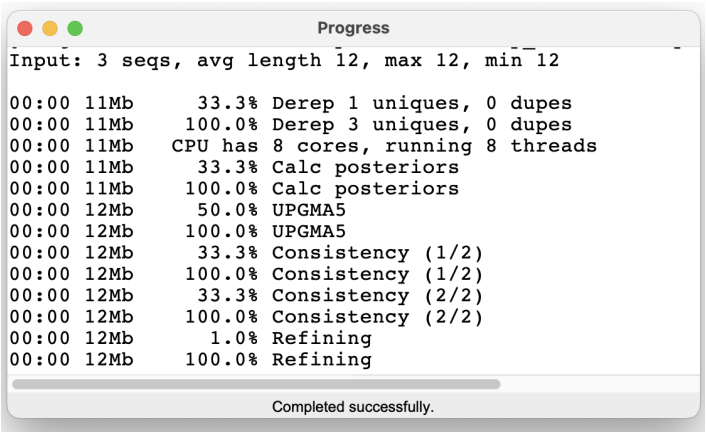


Figure 3.13. Muscle tool progress window.

3.2.6.4. Manual Alignment. Manual Alignment allows users to refine sequence alignments by directly editing them using various operations, such as shifting positions, inserting gaps, or deleting gap-only columns. This feature is crucial for correcting

alignment errors and ensuring accuracy in cases where automated alignment tools might misplace residues, particularly in complex or biologically significant regions. The following edit operations, inspired by AliView's design, are available for manual sequence alignment:

- Move Selected Position Right: Moves the selected letters to the right if there is a gap available at the right position.
- Move Selected Position Left: Moves the selected letters to the left if there is a gap available at the left position.
- Insert Gap Move Right: Inserts a gap at the current selection position and shifts the selected sequences to the right.
- Insert Gap Move Left: Inserts a gap at the current selection position and shifts the selected sequences to the left.
- Replace Selected With Gap(s): Replaces the selected letters with gaps.
- Delete Gap-Only Columns: Deletes columns that consist solely of gaps.

These operations allow for manual adjustments and refinements of sequence alignments, making it easier for users to manipulate sequences as needed for their analysis.

3.2.7. Annotation

The annotation component of the app allows users to view and modify sequence annotations. Annotations in sequence files provide additional information about the sequence, such as functional features, structural elements, and metadata, which can enhance the understanding and analysis of the biological data. The platform supports annotations from a range of formats. It features a dedicated annotation window, providing users with the ability to edit various types of annotations, including general information, key-value pairs, feature table, letter annotations, and cross-references to external databases. This functionality offers a comprehensive solution for sequence annotation.

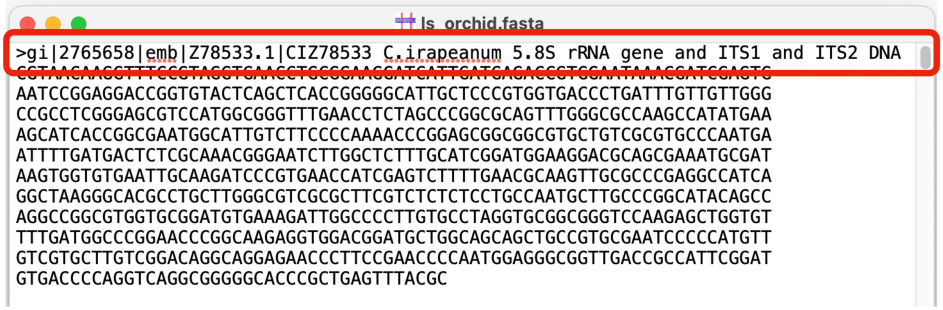


Figure 3.14. An example FASTA file.

Annotation support varies across file formats. Simple formats like FASTA as shown in Figure 3.14 have limited annotations, while formats like GenBank, EMBL or UniProt-XML contain rich annotations as shown in Figure 3.15. On the other hand, FASTQ and Stockholm files support letter annotations. The annotation support for this app is a key advantage compared to most alternative tools.

The application has a dedicated annotation window which allows users to inspect and modify annotations of the selected sequence, as shown in Figure 3.16. The fields in this window are designed to align with Biopython's annotation object structure. The window has five tabs:

3.2.7.1. Information Tab. Displays general information about the sequence, as shown in Figure 3.16. User can edit these fields using menu items.

- Id: This field serves as the main identifier for a sequence. In most cases, it corresponds to an accession number. This primary ID allows users to uniquely identify sequences in the dataset. [Adapted from [38]]
- Name: A more familiar name or alternative identifier for the sequence. This could be a clone name or an alias and is analogous to the LOCUS field in GenBank records. In some instances, this name may overlap with the accession number. [Adapted from [38]]
- Description: A human-readable and detailed description of the sequence. This field provides additional expressive context. [Adapted from [38]]

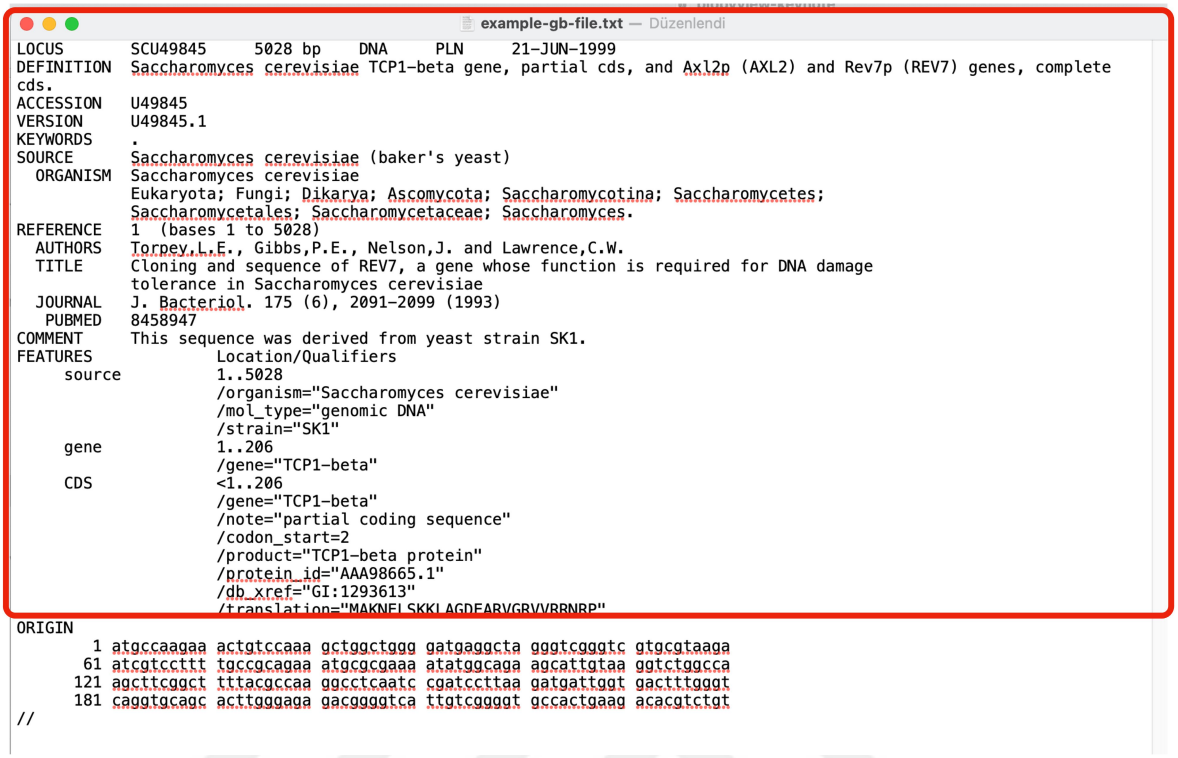


Figure 3.15. An example GenBank file.

3.2.7.2. Annotations Tab. A two-column table designed to store additional metadata about the sequence, as shown in Figure 3.17. The keys denote the type of metadata, while the values hold the corresponding information, providing flexibility for adding various types of unstructured details to the sequence. The table has a nested structure where an item may contain a set of inner items. We designed it to match the structure used in corresponding file formats (e.g., GenBank, EMBL, UniProt-XML) to accurately represent their information. Users can add, delete, or edit rows at any level, where each row consists of a key-value pair. [Adapted from [38]]

The annotations displayed in this window correspond to those shown in an example GenBank file in Figure A.1. In Figure A.2, these annotations are highlighted to illustrate how they appear in the application, showing the mapping between the file structure and the app's interface through color coding. The red area contains information such as molecule type, accession numbers, and keywords. The green area displays organism and taxonomy details, while the blue area shows relevant publications. All information is presented in a nested list structure, displaying every detail down to its

sublevels. This allows users to view, add, remove, or edit data at any desired level, providing flexibility for users.

The annotations shown in Figure A.3 correspond to the reference section in the GenBank file, which lists related publications. Figure A.4 illustrates how these references are displayed in the app. Each reference includes author, title, journal, and location information. Since a reference may be linked to a specific sequence fragment, the location is recorded separately. All information in this section is fully editable.

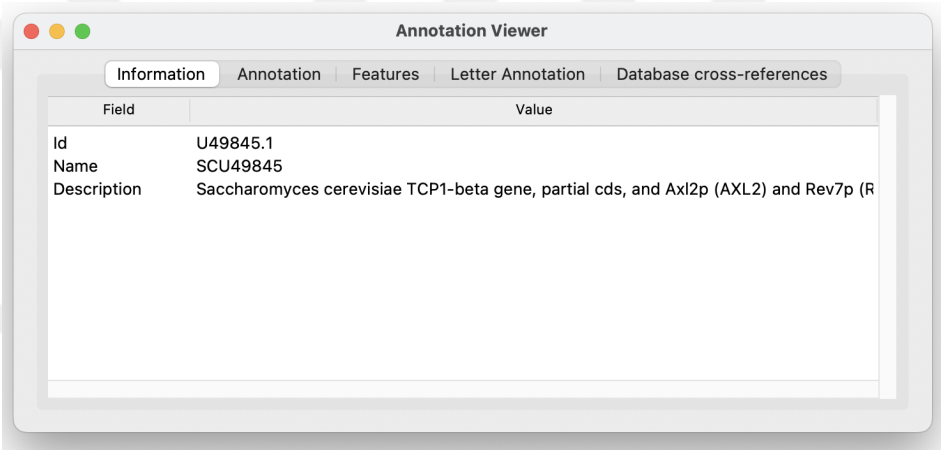


Figure 3.16. Information tab.

3.2.7.3. Features Tab. A two-column table designed to store feature table elements available in formats such as GenBank or EMBL, as shown in Figure 3.18. Each item belongs to a feature which is an annotation related to a specific region in the sequence. The keys denote the properties of the respective feature, while the values hold the corresponding information. Users can modify rows. [Adapted from [38]]

A sequence typically contains multiple features, each representing a biologically significant function or characteristic. Figure A.5 shows how features are displayed in a GenBank file, while Figure A.6 illustrates how they are presented in the application. All feature table elements and their attributes are fully extracted.

Figure A.7 shows CDS, a common feature representing a coding sequence—the DNA segment that translates into a protein. Figure A.8 illustrates how CDS is displayed in the app. Like other features, CDS includes a location and various qualifiers such as note, function, and translated sequence, all of which are editable. Compared to other features, location is more complex. The location editing interface, shown in Figure 3.19, allows users to modify properties like start, end, and strand with selectable options. This ensures full compatibility with feature table formats in GenBank, EMBL, and UniProt-XML, allowing users to import, edit, or create features from scratch.

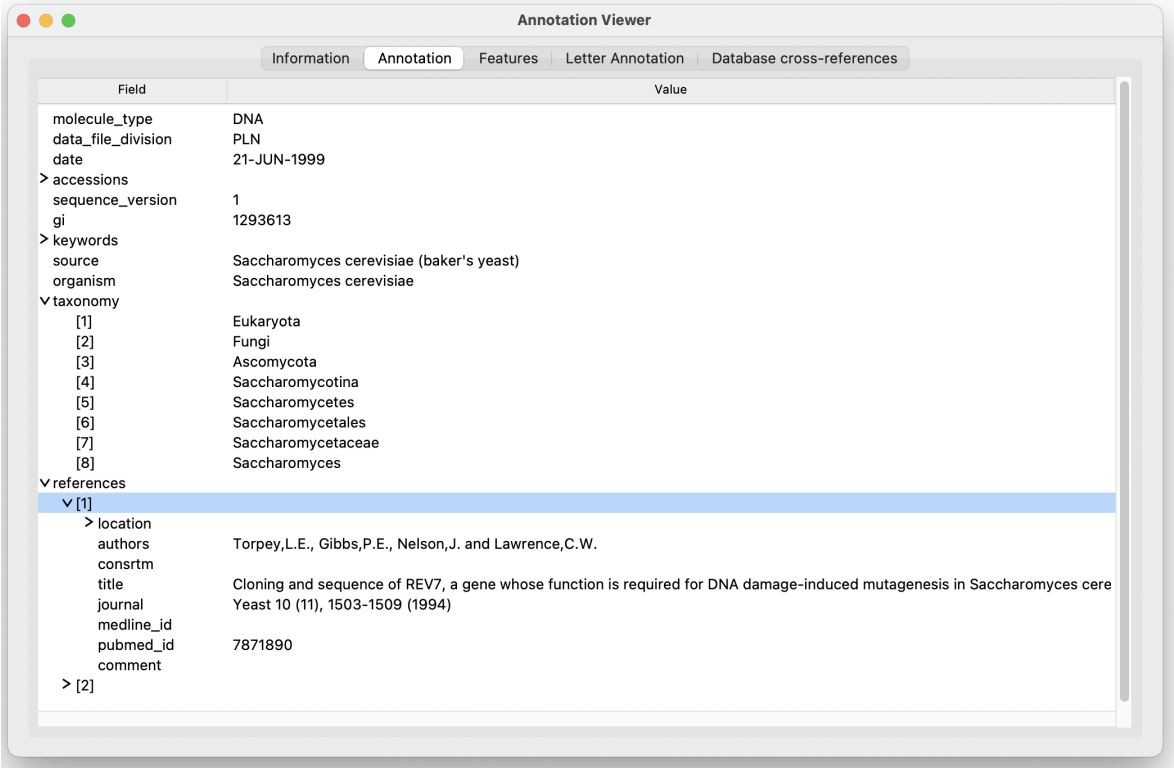


Figure 3.17. Annotation tab.

3.2.7.4. Letter Annotation Tab. Displays per-letter annotations, such as quality scores (from FASTQ files) or secondary structure data (from Stockholm or PFAM alignment files). Users can add, delete, or edit rows and columns. Columns correspond to annotation names, and rows correspond to each letter's position in the sequence, as shown in Figure 3.20. [Adapted from [38]]

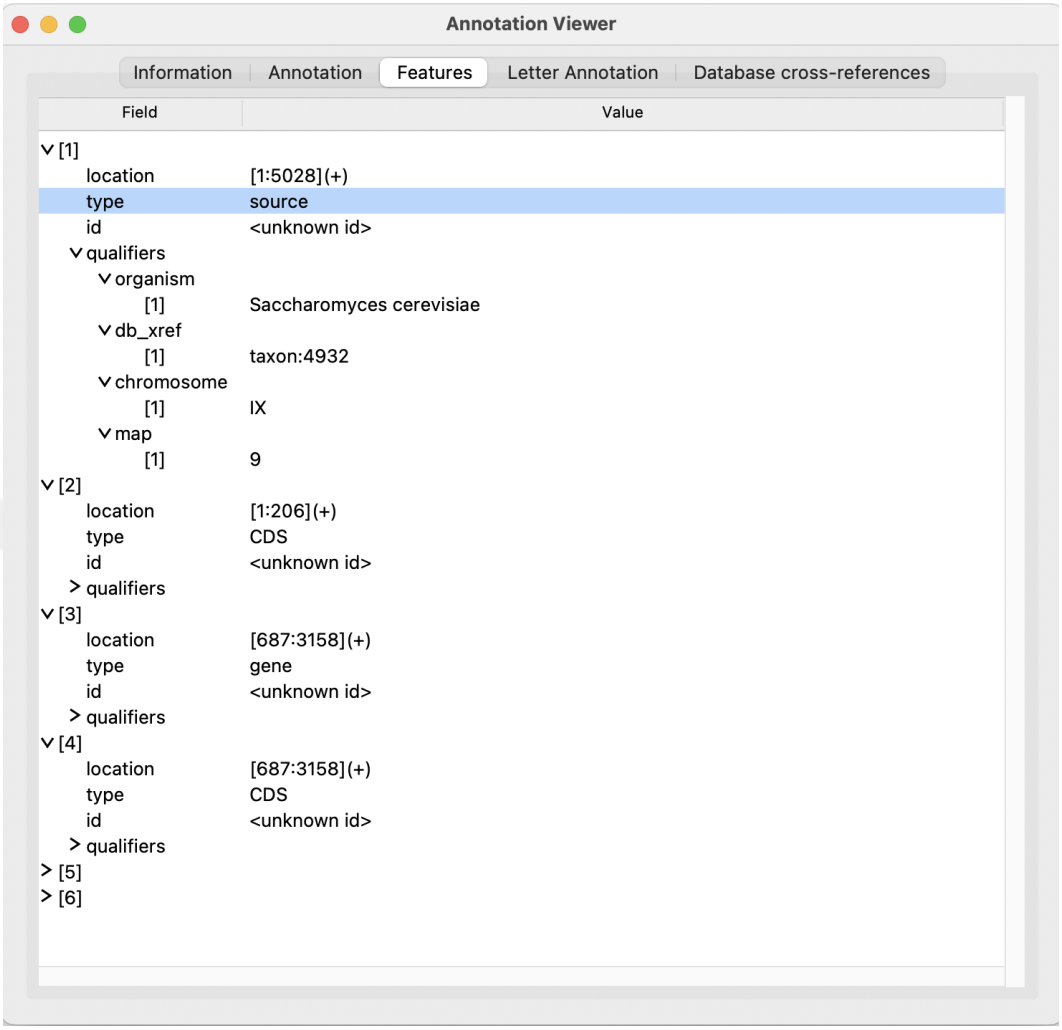


Figure 3.18. The app's features tab.

3.2.7.5. Database Cross-References Tab. A list of cross-references to external databases, represented as rows of a one-column table, as shown in Figure 3.21. These references help link the sequence to external resources for further exploration or verification. Users can add, delete, or edit rows, with each row representing a cross-reference. [Adapted from [38]]

After editing the desired fields for a specific sequence, the user can choose to save annotations . It is important to note that, unlike the edits in sequence viewer, changes made in the annotation window cannot be undone after saving, so users should ensure the edits are correct before saving.

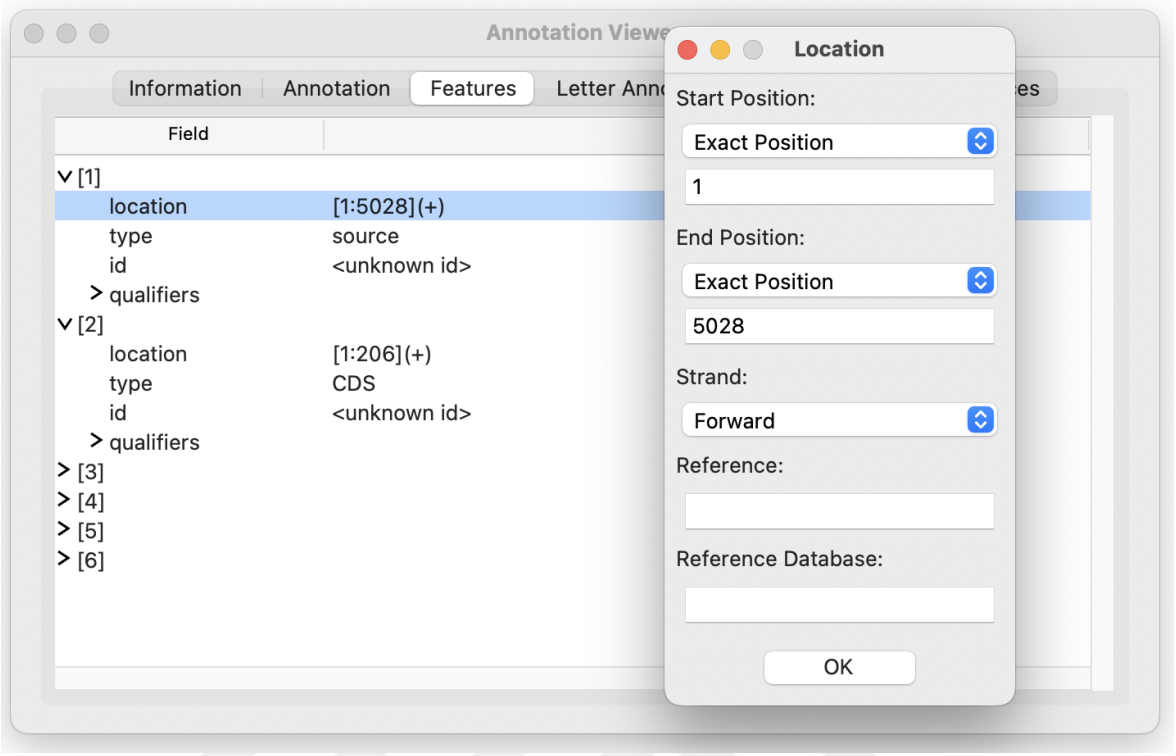


Figure 3.19. Editing location in features tab.

3.2.8. Phylogenetic Tree

This component provides tools for parsing, visualizing, and inferring phylogenetic trees from sequence data. It includes features for opening existing trees, performing tree inference from sequence alignments, and converting between various tree file formats. These features are integrated with external tools like FigTree, FastTree, and RAxML, and offer a straightforward interface for working with phylogenetic data. The progress of the tree tools is displayed in real-time within the application, similar to the alignment tools, ensuring users are kept informed during the process. The component also supports built-in functionality for format conversion using Biopython.

3.2.8.1. Open Tree. Supports parsing and visualization of phylogenetic trees using FigTree. We have an interface to this program, similar to those for alignment tools, with access to the most basic parameters. Figure 3.22 shows the window for running this feature.

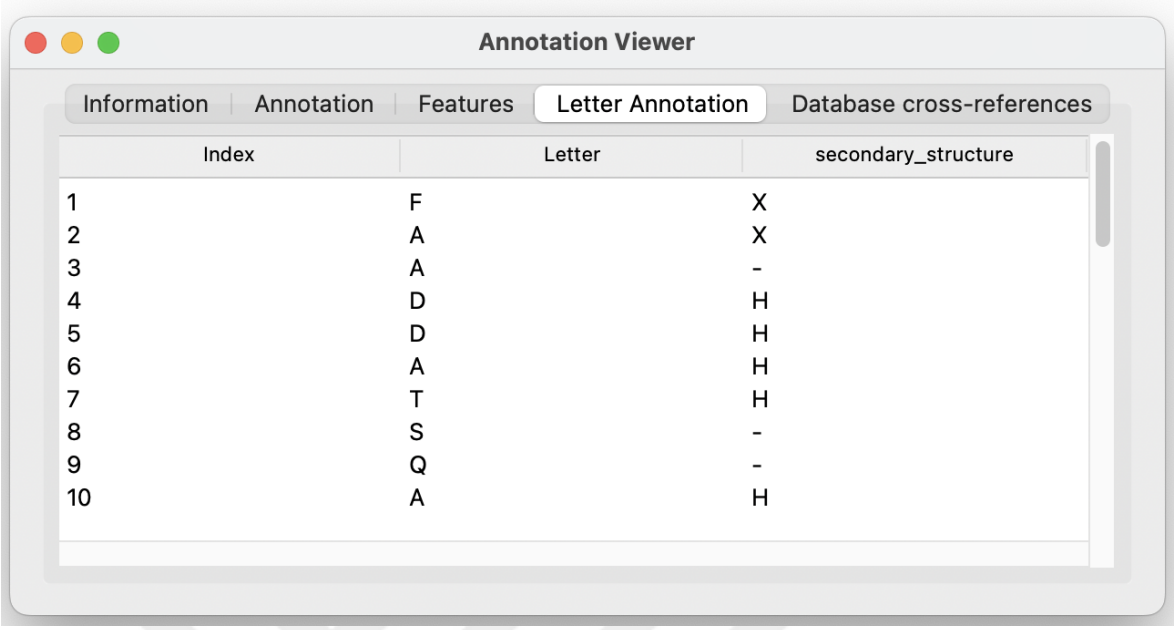


Figure 3.20. Letter annotation tab.

3.2.8.2. Infer Tree. Allows tree inference from alignments using external tools like FastTree and RAxML. We have interfaces to these programs, as well. Figure 3.23 shows the window for running FastTree.

3.2.8.3. Convert File Format. Converts between tree file formats. Supported formats include `newick`, `nexus`, `phyloxml`, and `nexml`. This is a built-in feature using Biopython functions, not an external program. Figure 3.24 shows the window for running the conversion.

3.2.9. Motif

Motif search helps identify biologically significant patterns in sequences. The Motif component provides tools for creating, reading, editing, saving, and searching motifs within biological sequence data. It supports both instance-based and count matrix-based motif creation, allows detailed motif analysis through position-weight matrices and scoring matrices, and includes options for comprehensive motif search. It supports two main types of searches: Exact Search, which looks for perfect matches of a given pattern, and Approximate Search, which allows for variations to account for

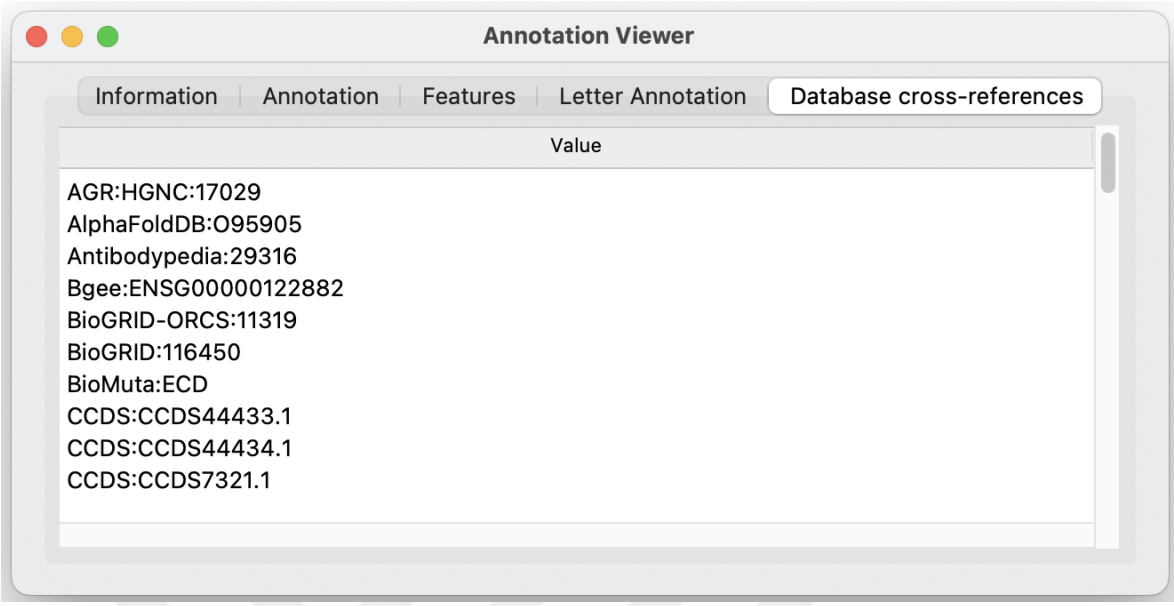


Figure 3.21. Database cross-references tab.

mutations and biological flexibility. The component also offers flexibility in adjusting pseudocounts and background probabilities for more refined analysis.

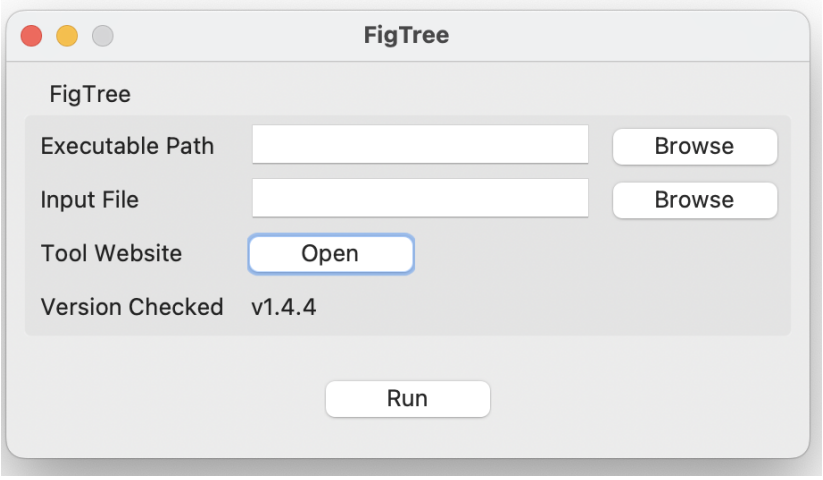


Figure 3.22. FigTree window.

3.2.9.1. Create Motif. Motifs can be created (or edited later) by the user using either motif instances or a count matrix:

Motif Instances: This option allows motifs of varying sizes to be created. Instances are entered in a defined text editor, as shown in Figure 3.25.

Count Matrix: A fixed-size motif group can be created by specifying a count matrix. Each row represents a position in the motif and contains count values for each letter in the alphabet (DNA, RNA, or Amino acids). The UI for this feature is shown in 3.26.

3.2.9.2. Read Motif. Instead of creating motifs manually, the application allows users to read motifs from files. Supported motif formats include: AlignAce, ClusterBuster, XMS, MEME, MINIMAL, MAST, TRANSFAC, pfm-four-columns, pfm-four-rows, pfm, jaspar, and sites. Some motif files contain instances, while others use a count matrix. Read motifs can be edited using both instances and the count matrix.

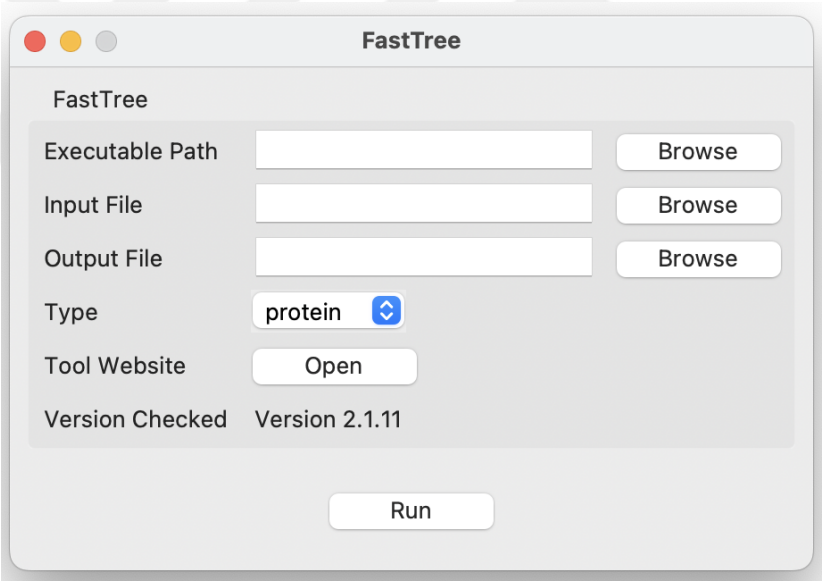


Figure 3.23. FastTree window.

After creating or reading motifs, the motif window displays the details of the currently selected motif in four tabs: Instances, Counts, Position-Weight Matrix (PWM), and Position-Specific Scoring Matrices (PSSM). Additionally, the window includes menu items for adjusting Pseudocounts and setting Background Probabilities.

Instances Tab: This tab displays motif instances in a text editor. However, some read motifs may not include instances and only contain a counts matrix.

Counts Tab: Displays how often each nucleotide appears at each position along the alignment, within a table. An example count matrix is shown in Figure 3.27. Only enabled when we have fixed length motifs.

Position-Weight Matrix (PWM) Tab: Displays the normalized matrix, where each value represents the probability of a nucleotide at a specific position. PWM matrix for the count matrix in Figure 3.27 is shown in Figure 3.28. Only enabled when we have fixed length motifs.

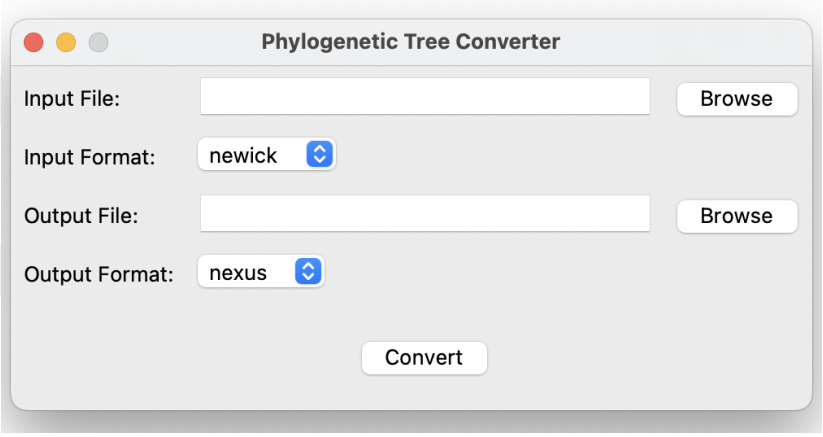


Figure 3.24. Tree convert window.

Pseudocounts: The window allows the user to adjust pseudocounts, which are added to each position in the matrix before normalization. This prevents overfitting of the PWM to the limited number of motif instances and ensures probabilities do not become zero [38].

Position-Specific Scoring Matrices (PSSM) Tab: PSSM tab shows log-odds ratios for each symbol, indicating the likelihood of a symbol belonging to the motif versus the background: positive values signify higher frequency in the motif, negative values indicate higher frequency in the background, and a value of 0 means the symbol is equally likely in both [38]. PSSM matrix for the PWM in Figure 3.28 is shown in Figure 3.29. Only enabled when we have fixed length motifs.

Background Probabilities: Users can define unequal probabilities for letters in the alphabet to calculate the PSSM against a specific background.

Save Motif: Motifs created from scratch or read from a file and subsequently edited can be saved, but only fixed-length motifs are supported, as motif formats require this restriction. Supported formats include clusterbuster, pfm, jaspar, and transfac. Among these, clusterbuster, pfm, and jaspar are limited to DNA motifs only and pfm format do not support multiple motifs.

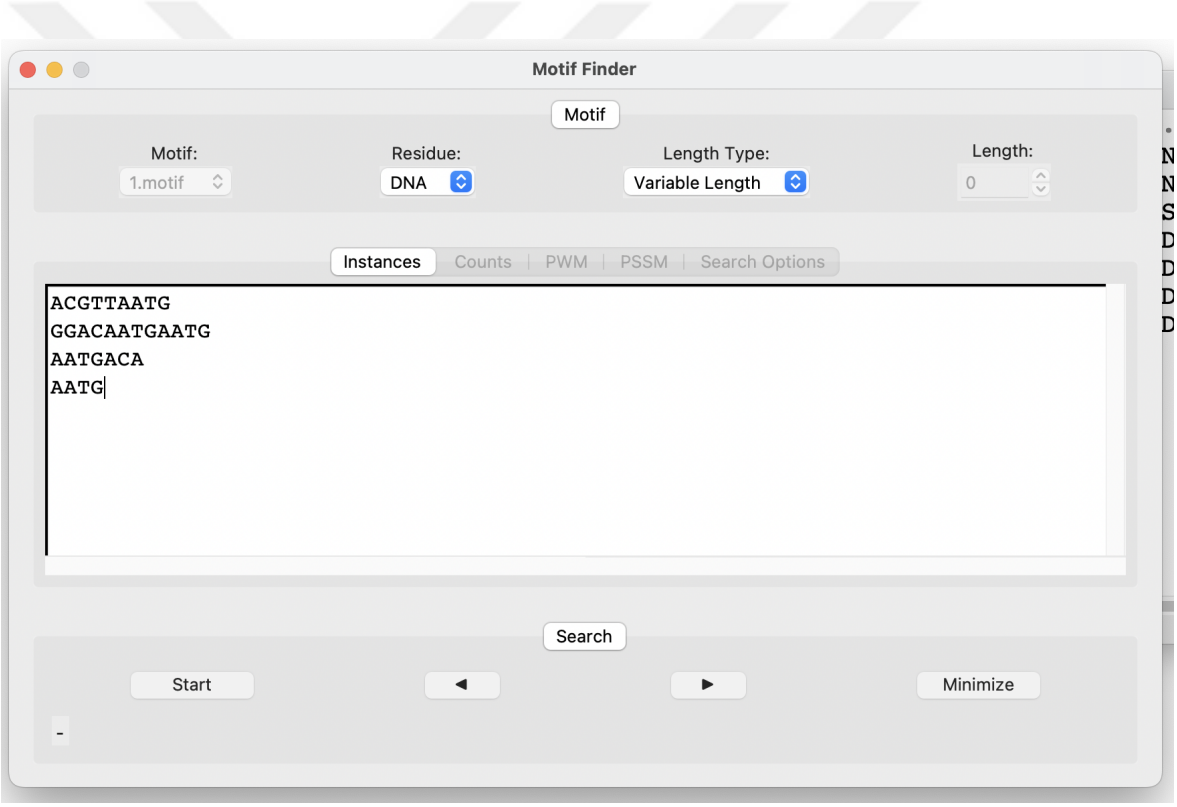


Figure 3.25. Motif creation using instances.

Motif Search: The application provides a detailed interface for searching patterns within the currently active dataset or file, as shown in Figure 3.31. It includes various search options and supports two types of searches: Exact Search and PSSM-Based Search. The search is performed using separate Python threads to ensure the user interface remains responsive and to provide real-time progress updates. Searches are conducted sequentially, with "Next" and "Previous" buttons for navigation. The "Next" button initiates a new thread, continuing the search from where the last match

was found, while the "Previous" button iterates through already found matches, making it instantaneous. Additionally, the window features a minimize button, allowing users to reduce the panel and view the sequence viewer alongside the search interface during the search process, which is shown in Figure 3.30.

3.2.9.3. Search Options. The search window, as shown in Figure 3.31, offers flexibility with its configurable options. The user can define the search scope, choosing between searching in all sequences or restricting the search to selected sequences. Additionally, if the "search in all sequences" option is selected, the search start point can be specified, allowing the search to either start from the top of the sequence list or begin from the currently selected sequence.

Exact Search: Exact search scans sequences in the selected area one by one to find all instances that perfectly match any given motifs, regardless of their length. This problem, known as the Multiple Pattern Matching Problem, involves searching for several patterns at once. Biopython employs a Brute Force Approach to identify these matches efficiently. A key application of multiple pattern search is Read Mapping, where sequencing reads are aligned to a reference genome by identifying regions with exact matches.

PSSM-Based Search: PSSM-based search is a type of approximate search that identifies regions resembling a group of motifs as a whole. Unlike exact search, it accounts for mutations by finding approximate matches rather than requiring an exact sequence match. Users can specify a similarity threshold, determining how closely a sequence must resemble the motif group to be considered a match. This method is widely used in Transcription Factor Binding Site (TFBS) Prediction, where it helps identify regulatory DNA regions, and in Homology Search within tools like BLAST to detect evolutionary relationships between sequences.

3.2.10. Other Tools:

Tools added in the settings window are listed under this menu with the given names, providing easy access to frequently used external programs.

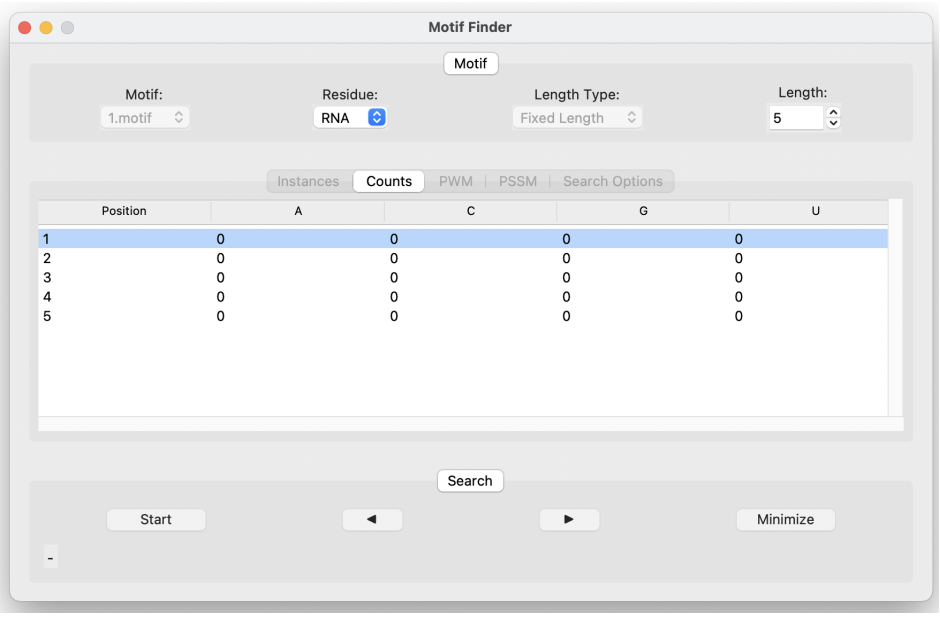


Figure 3.26. Motif creation using count matrix.

The Other Tools section offers a versatile interface that enables users to execute external bioinformatics tools directly from within the application. Users can input command-line prompts into a dedicated input field.

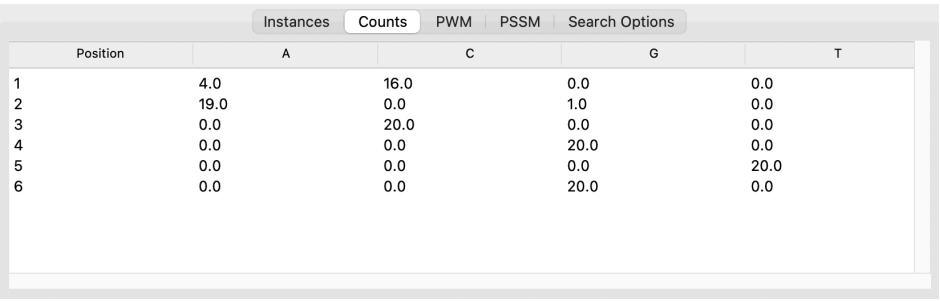
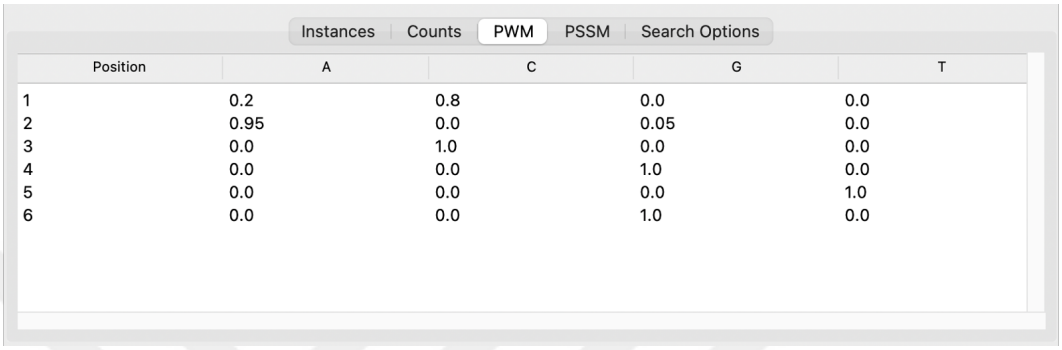


Figure 3.27. Count matrix.

Upon execution, Biopyview initiates the specified command using a subprocess, captures its output, and displays the real-time progress in a dynamic, scrollable text

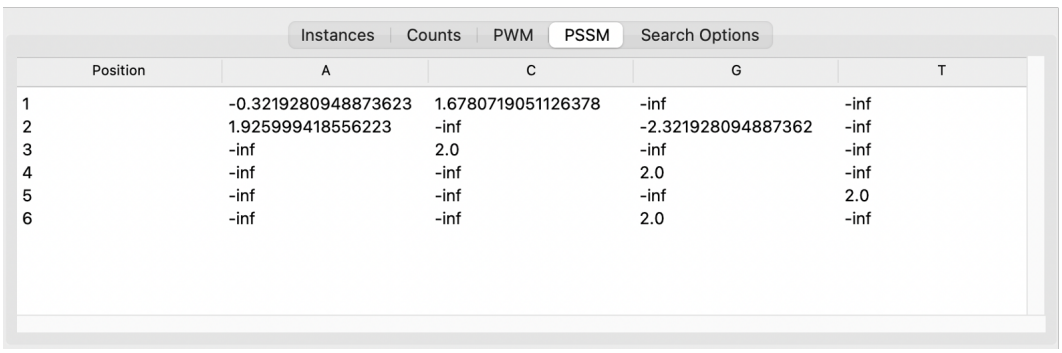
area. This live output stream provides immediate feedback, allowing users to monitor long-running tasks, check for potential errors, and verify intermediate results without leaving the graphical environment.



Position	A	C	G	T
1	0.2	0.8	0.0	0.0
2	0.95	0.0	0.05	0.0
3	0.0	1.0	0.0	0.0
4	0.0	0.0	1.0	0.0
5	0.0	0.0	0.0	1.0
6	0.0	0.0	1.0	0.0

Figure 3.28. PWM matrix.

Once the execution finishes, the interface clearly indicates whether the process completed successfully or encountered an error, using a status label. This system provides significant convenience for users who frequently run specific command-line tools such as alignment programs, tree builders or format converters.



Position	A	C	G	T
1	-0.3219280948873623	1.6780719051126378	-inf	-inf
2	1.925999418556223	-inf	-2.321928094887362	-inf
3	-inf	2.0	-inf	-inf
4	-inf	-inf	2.0	-inf
5	-inf	-inf	-inf	2.0
6	-inf	-inf	2.0	-inf

Figure 3.29. PSSM matrix.

By integrating this functionality into the graphical user interface, Biopyview reduces the need to switch between terminal windows and the main application, thereby streamlining the workflow. Furthermore, users can copy or save the output for downstream analysis or reproducibility purposes.

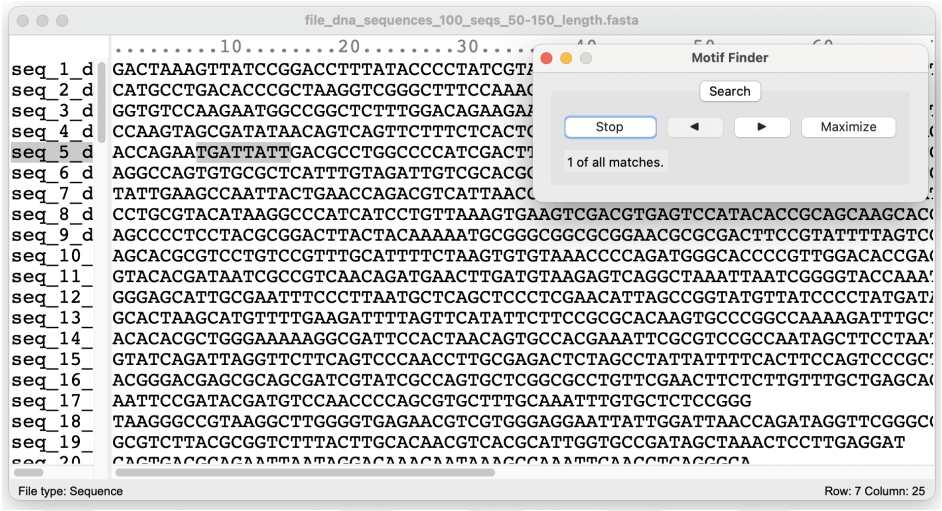


Figure 3.30. Minimized search window.

This feature is particularly useful for advanced users who have established command-line pipelines or who prefer the flexibility of custom tool usage, all while benefiting from the convenience of real-time monitoring and clear status reporting within the familiar Biopyview environment.

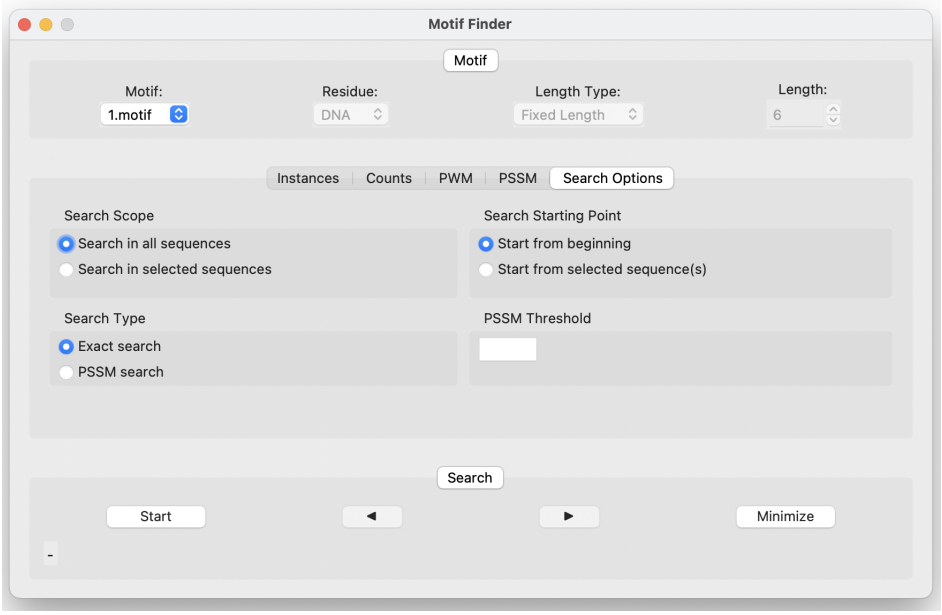


Figure 3.31. Motif search options.

4. RESULTS AND EVALUATION

This chapter presents a comprehensive evaluation of the proposed tool, focusing on its performance metrics, memory efficiency, and functionality in comparison to existing alternatives. The evaluation includes a comparison with four prominent tools in this domain: Jalview, Seqotron, Seaview, and AliView. These tools are well-established and widely adopted in the field, providing a solid benchmark for assessing the effectiveness of our tool.

4.1. Functionality Comparisons with Existing Alternatives

In this section, we compare the functionality of the proposed tool with mentioned alternatives to highlight its advantages and unique features.

4.1.1. Built on Biopython

The proposed platform is built on Biopython, a versatile and widely supported library for biological computations. Biopython consists of numerous packages that improve continuously, offering extensive bioinformatics functionalities. Our research indicates that there is currently no biological sequence analysis GUI tool developed using Biopython. By utilizing Biopython, our tool achieves rapid implementation and provides access to continuously updated bioinformatics features, whereas other tools rely on their standalone implementations. While some tools use bioinformatics libraries or external tools for specific tasks, such as reading certain file types, they generally parse files and perform computations using their own implementations.

Moreover, Biopython's active development enable our platform to add additional functionalities and support more file formats in the future, ensuring adaptability to the evolving needs of researchers.

By incorporating features from six out of the seventeen chapters of Biopython documentation into the current version, our platform lays a foundation for future expansion, allowing for the seamless integration of additional features as needed.

4.1.2. Wide File Format Support

Thanks to Biopython's comprehensive file parsers, our tool can support opening a broad range of file formats, surpassing the flexibility of existing tools. These files can contain both unaligned and aligned sequences. The number of supported file formats for our tool and others is as follows:

- BioPyView: 38
- Jalview: 19
- Seqotron: 8
- Seaview: 6
- AliView: 6

In addition to supporting significantly more file formats than other tools, we also identified the 10 most commonly used file formats and compared the support provided by our app and existing applications in Table 4.1. While our app supports all 10 formats, others support far fewer: Jalview supports 4/10, Seaview and AliView support 3/10, and Seqotron supports 2/10. Jalview's GenBank support is limited because it can only open a single sequence from the file, whereas GenBank files typically contain multiple sequences. This restriction reduces its usability for handling full datasets and represents a key limitation compared to more comprehensive tools. To wrap up, while our app can handle all commonly used formats, users of other tools would need to rely on multiple applications to work with different file formats.

Additionally, there are plenty of formats available exclusively in our app. These formats are used for analysis in different subfields of bioinformatics and are outputs from related databases and software. A table of these formats and their definitions is

provided in Table 4.2. This table is adapted from [33] and [34] and a more extensive version can be found there.

Table 4.1. File format support comparison table.

File Format	**BioPyView**	**Jalview**	**SeaView**	**AliView**	**Seqotron**
GenBank	✓	Limited	X	X	X
EMBL	✓	X	X	X	X
UniProt XML	✓	X	X	X	X
Swiss-Prot	✓	X	X	X	X
FASTQ	✓	X	X	✓	X
PIR	✓	✓	X	X	X
Stockholm	✓	✓	X	X	✓
MSF	✓	✓	✓	✓	X
Nexus	✓	X	✓	✓	✓
Mase	✓	X	✓	X	X
TOTAL	10/10	4/10	3/10	3/10	2/10

4.1.2.1. Reading Files with Multiple MSAs. Our application effectively utilizes Biopython's alignment file parsers, which enable seamless reading of files containing multiple alignments. This capability is particularly beneficial when dealing with datasets that include resampled alignments, such as those generated by the PHYLIP tool `seqboot`, or multiple pairwise alignments from EMBOSS tools like `water` and `needle`, as indicated by Biopython tutorial [38]. In contrast, alternative tools exhibit limitations in their ability to process multiple MSAs:

- Aliview: When tested with PHYLIP files, Aliview only opened the first alignment, restricting its utility for comprehensive analyses.
- Seqotron: In trials using the Stockholm format, Seqotron stacked alignments without proper separation, complicating the interpretation of results.

Table 4.2. File formats only available in our app.

File Format	**Description**
maf	Store whole-genome alignments from UCSC genome browser.
mauve	Alignment file format of Mauve software.
Ace	Contains contig sequences (results of genome assembly).
abi/abi-trim	Sequences from Sanger sequencing machines that include raw sequence data and quality scores.
sff/sff-trim	Sequences from older sequencing technologies like Roche 454 and IonTorrent.
phd	Created by PHRED software, which assigns quality scores to DNA bases and is used in assembling DNA sequences.
imgt	A format used for storing annotated immune system-related DNA or protein sequences.
seqxml	A simple XML-based format for sharing sequence data between tools.
qual	Files that store quality scores but do not include the DNA sequence itself.
tab	Simple two-column tab-separated sequence files. This is used by Aligent's eArray software.
gck	A format used by Gene Construction Kit software for designing DNA sequences, such as plasmids.
snapgene	A format used by SnapGene software, popular for visualizing and designing genetic constructs.
xdna	A format from DNA Strider and Serial Cloner software for storing and editing DNA sequences.

- Jalview: Similar to Aliview, Jalview was limited to processing only the first alignment when working with Stockholm files.

- Seaview: This tool also demonstrated a limitation by opening only the first alignment when tested with PHYLIP files.

These shortcomings underscore the strengths of our tool in handling alignment files containing multiple MSAs.

Obviously, the idea of developing the application based on Biopython has been significantly beneficial in terms of format support. While other apps have been developed over many years and the format support has been enhanced as new versions emerge, ours can support far more formats than any of them. This wide file format support ensures compatibility with diverse input types, making our platform much more versatile compared to existing alternatives.

4.1.3. Annotation Viewer and Editor

Our platform includes an integrated annotation viewer and editor that allows users to handle files containing aligned or unaligned protein or nucleotide sequence data along with annotations. Our app supports key annotation-rich file formats, including GenBank, EMBL, and UniProt-XML. GenBank, sourced from the NCBI GenBank database in the USA, contains DNA and RNA sequences. Similarly, EMBL, provided by the European Nucleotide Archive (ENA) in Europe, also stores DNA and RNA sequences. EMBL and GenBank are part of the INSDC, a global collaboration that collects and shares sequences worldwide. As they are primary nucleotide databases, supporting these formats is crucial for comprehensive sequence analysis and interoperability. Additionally, UniProt-XML, sourced from the UniProt database, contains protein sequences and is one of the most common protein databases available. Our app fully supports these formats, extracting all annotations and providing editing, viewing, and saving capabilities in dedicated interfaces. These features are explained in detail in Section 3.2.7. An overview of the annotation window is seen in Figure 4.1. Specifically, our tool presents annotations in five tabs:

- Information Tab: Displays general sequence details (ID, Name, Description). Editable via menu items.
- Annotations Tab: Stores additional metadata in a two-column table with a nested structure and designed to match formats like GenBank, EMBL, and UniProt-XML. Users can add, delete, or edit key-value pairs at any level.
- Features Tab: Stores and edits feature table elements (e.g., CDS, gene). Fully supports GenBank, EMBL, and UniProt-XML.
- Letter Annotation Tab: Handles per-letter annotations (e.g., quality scores, secondary structures). Users can modify rows and columns.
- Database Cross-References Tab: Lists external database links, allowing users to add, edit, or remove references.

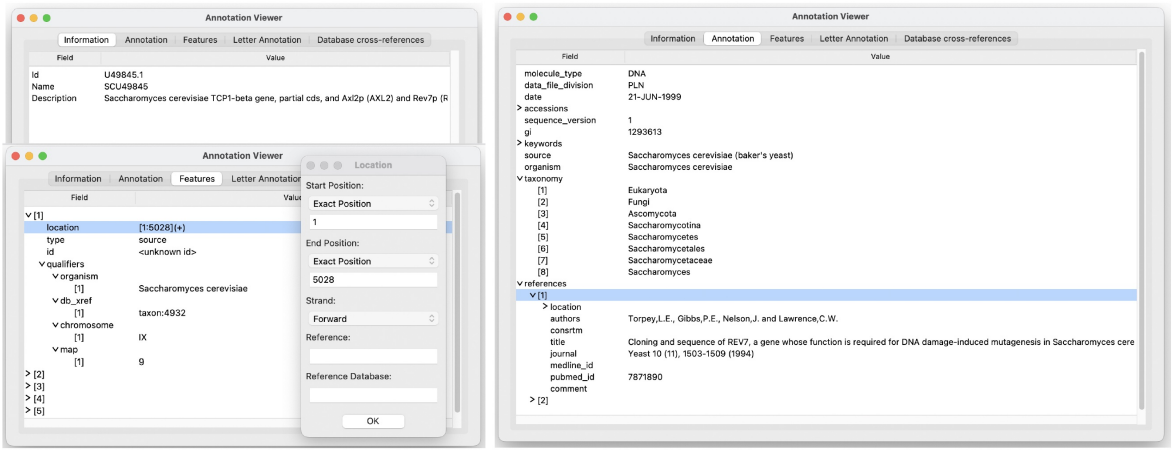


Figure 4.1. Annotations windows.

In comparison, other tools offer very little annotation support. The support for the three annotation-rich formats—GenBank, EMBL, and UniProt-XML—across other applications is presented in Table 4.3. As seen, only Jalview supports GenBank, and even that support is highly limited. None of the tools support the other two formats, EMBL and UniProt-XML. The annotation support is explained in detail below.

- Jalview: Jalview provides limited support for annotation handling, allowing users to open and edit GenBank files partially. It can retrieve only a single sequence from a GenBank file. If the file contains multiple sequences, which is common, it

Table 4.3. File format support comparison table for annotations-rich formats.

File Format	**BioPyView**	**Jalview**	**SeaView**	**AliView**	**Seqotron**
GenBank	✓	Limited	X	X	X
EMBL	✓	X	X	X	X
UniProt XML	✓	X	X	X	X

cannot process the others. Further, it only retrieves very basic annotations and restricts editing to sequence name and description. While users can save their modifications, the output is in Jalview's custom format rather than GenBank, making it incompatible with other bioinformatics tools. This limitation reduces interoperability and prevents seamless integration with widely used sequence analysis pipelines. On the other hand, Jalview lacks support for per-sequence letter-based annotations , such as secondary structure or quality scores. For instance, Jalview can only parse consensus-level secondary structure information from Stockholm files, while our tool supports per-sequence secondary structure information.

- Seaview: Provides limited support for annotations. It allows importing certain unstructured sequence-based annotations, such as "comments," and alignment-based annotations like "footers," but only in specific formats (e.g., Mase and Nexus).
- AliView and Seqotron: Both tools only handle sequence names and sequences themselves, without any support for annotations.

In summary, our application fully supports annotation-rich formats like GenBank, UniProt-XML, and EMBL, reading and displaying all annotations in detail. It also processes letter-based annotations and other metadata from simpler formats. Annotations are presented in comprehensive, structured windows with full editing and saving capabilities. In contrast, Jalview handles only basic annotations, Seaview supports comments, and other tools lack annotation support entirely. Our annotation handling is significantly more advanced than existing alternatives.

4.1.4. Extensive Pattern Search

The proposed tool offers advanced pattern search capabilities in a dedicated window that are not available in existing alternatives. It supports both exact and PSSM-based approximate pattern matching, provides searching for specific regions, and allows users to open and save common pattern files. Furthermore, the ability to perform multiple pattern matching further enhances the functionality, enabling researchers to conduct detailed sequence analyses efficiently.

In contrast, other apps have very basic search interfaces. Seaview offers a simple search option, identifying patterns across gaps during the search process. Aliview can also identify patterns across gaps and additionally adheres to IUPAC codes to search for ambiguous patterns. Seqotron only includes a simple search button. Finally, Jalview offers regex-based search, enabling users to search within both annotations and sequences.

These comparisons highlight that while other tools may offer some basic search functionalities, they are constrained to single pattern matching and do not provide the comprehensive features of our proposed tool such as handling pattern files, supporting multiple pattern search and PSSM based approximate matching.

4.1.5. Index Mode: Handling Very Large Files

One of the significant advantages of our tool is its ability to handle files of enormous size. This is achieved through an efficient indexing process, where sequences in the file are initially indexed and only cached in memory when they are viewed. This approach prevents memory overload and allows researchers to process extremely large files seamlessly. Among the compared tools, only our platform and Aliview implement this feature. Other tools lack this feature and can only open files that fit into available RAM. The memory usage results are discussed in Table 4.6.

The memory and speed results for two 10 GB datasets opened in index mode are shown in Table 4.4. These datasets differ in dimensions: one contains 100,000 sequences, each with 100,000 characters, while the other contains 1 million sequences, each with 10,000 characters. As shown in the table, both datasets took around 1.5 minutes to load. The memory usage for one was 100MB and the other 200MB, which is similar to the memory consumption when the app is first launched. This indicates that very little additional memory is used. Normally, opening 10GB datasets on a PC with 16GB of RAM—like ours—would not be possible, as we typically use around 3GB of memory even for datasets as small as 1GB, which is the best among the alternative tools, as seen in Table 4.6. This feature enables the opening of very large files without straining memory. Additionally, since Aliview implements index mode slightly differently, we did not include it in the comparison to ensure fairness.

Table 4.4. Index mode results.

Dimension Size	**Indexing Time**	**Memory**
100K x 100K	1.5 min	100 MB
1M x 10K	1.5 min	200 MB

4.1.6. Cross-Platform Compatibility

Unlike Seqotron, which is limited to specific operating systems, our tool and other alternatives like Jalview, Seaview, and Aliview are cross-platform and work seamlessly on multiple operating systems. This ensures broader accessibility and usability for researchers across different platforms.

4.2. Memory Usage Evaluation

This section evaluates the memory efficiency of the proposed tool relative to widely used alternatives in the field. Memory usage tests were performed across various scenarios, including different file sizes and conditions.

4.2.1. Testing Methodology

This command reports the Resident Set Size (RSS). We chose to use RSS as the metric for memory usage because it provides an accurate representation of the physical memory occupied by a process in main memory (RAM). Unlike virtual memory, which includes memory allocations that may not actually be loaded into RAM, RSS measures only the portion of memory actively held in RAM. This distinction is critical for evaluating tools that process large datasets, as it reflects the actual RAM usage during operation. Other portions of a program's memory, such as those paged out to swap space or those not yet loaded from the executable, are excluded from RSS. During our tests, no memory was swapped out, further ensuring that the RSS values accurately reflected each application's RAM usage. For our purposes, this makes RSS an ideal measure of efficiency when handling large sequence datasets, providing clear insights into each application's performance under typical conditions. To measure the memory usage, we utilized the Linux command:

```
ps -p <PID> -o rss
```

We created mock DNA sequence files in FASTA format with varying sequence counts and sequence lengths, ranging in size from 1 MB to 1 GB. Each test file was designed to represent different levels of complexity and scale, ensuring a comprehensive evaluation of the application's performance. Specifically:

- Sequence Count: Files were generated to include anywhere from a few sequences (e.g., 10 or 100) to millions of sequences (e.g., 1M or 10M).
- Sequence Length: Sequence lengths varied from short sequences (e.g., 100 bases) to much longer ones (e.g., 100,000 bases).
- File Size: The resulting file sizes ranged systematically from 1 MB to 1 GB, representing small, medium, and large datasets commonly encountered in bioinformatics.

The test cases can be found in Table 4.5, which provides an overview of the generated mock DNA sequence files.

Table 4.5. Test cases for DNA sequences.

Number of Sequences	**Sequence Length**	**File Size**
10K	100	1.2 MB
100K	100	12.3 MB
1M	100	123.9 MB
1K	1K	1 MB
10K	1K	10.4 MB
100K	1K	103.9 MB
100	10K	1 MB
1K	10K	10.2 MB
10K	10K	101.9 MB
100K	10K	1.02 GB
10	100K	1 MB
100	100K	10.2 MB
1K	100K	101.7 MB
10K	100K	1.02 GB

These test cases were carefully designed to mimic real-world scenarios in bioinformatics, where researchers often work with datasets of diverse sizes and complexities. First, we measured the default memory usage of the applications before any file was loaded. Then we conducted tests in the following sequence for each file and application:

- Initial Measurement: After loading the entire file, we measured the memory usage, representing the memory consumed by the application once the file was fully loaded.
- Peak Test (After Interaction): After the file was fully loaded, we interacted with the content by scrolling up and down through the entire content. This allowed

us to measure the peak memory usage, representing the maximum memory consumed during typical user interactions.

4.2.2. Test Results

The Table 4.6 presents the memory test results, where the first rows indicate the initial memory measurements and the second rows represent the peak memory values observed after interaction. The interpretation of the results is as follows.

Table 4.6. Memory test results.

Test Case	**Biopyview**	**Aliview**	**Seaview**	**Seqotron**	**Jalview**
Startup	113,4	376,6	265,0	68,9	494,8
	-	-	-	-	-
10K x 100	133,2	556,6	245,2	94,4	585,3
(1.2 MB)	160,0	860,6	910,2	174,7	603,9
100K x 100	246,2	716,7	369,7	251,3	768,8
(12.3 MB)	267,9	970,1	1.147,1	387,5	817,8
1M x 100	1.317,7	2.110,3	1.600,4	1.704,1	2.584,0
(123.9 MB)	1.346,0	2.361,8	2.121,4	1.835,0	3.167,2
1K x 1K	120,4	501,8	264,2	141,1	644,1
(1 MB)	125,2	951,3	1.024,7	206,3	687,0
10K x 1K	160,9	544,0	264,2	188,1	851,5
(10.4 MB)	188,8	893,2	1.044,7	303,3	869,2
100K x 1K	549,5	909,4	540,3	1.113,7	1.903,8
(103.9 MB)	576,3	1.079,8	1.315,9	1.205,3	1.918,4
100 x 10K	141,7	521,9	239,5	104,9	556,9
(1 MB)	170,5	577,6	1.014,7	161,5	571,8
1K x 10K	155,5	523,1	266,1	195,4	807,9
(10.2 MB)	194,3	859,2	1.043,9	303,6	814,3

Table 4.6. Memory test results. (cont.)

Test Case	**Biopyview**	**Aliview**	**Seaview**	**Seqotron**	**Jalview**
10K x 10K	391,1	767,2	494,3	1.073,6	1.606,3
(101.9 MB)	424,1	1.044,0	1.271,7	1.187,2	1.678,4
100K x 10K	2.757,7	2.299,3	2.792,0	8.676,0	9.138,8
(1.02 GB)	2.791,7	2.636,7	3.569,7	9.307,6	9.152,2
10 x 100K	199,1	525,1	236,7	88,6	665,5
(1 MB)	-	-	-	-	-
100 x 100K	278,9	514,1	257,1	182,7	922,3
(10.2 MB)	443,7	673,7	924,4	246,3	942,3
1K x 100K	444,5	812,1	478,8	945,4	1.552,8
(101.7 MB)	720,2	1.002,4	1.256,4	1.064,1	1.568,3
10K x 100K	2.431,9	2.579,9	2.434,5	7.537,1	8.931,3
(1.02 GB)	2.800,8	2.972,8	3.212,1	7.673,8	8.944,7

For program startup, Seqotron performs best, followed by our tool. Jalview exhibits significantly higher values, which negatively impact the following results, as well. For 10K x 100, Seqotron performs best at initial score, but we outperform it in terms of peak values. In Seaview, the peak value nearly increases by four times compared to the initial value, which is typical for its behavior in most cases. Jalview's peak value is very close to the initial score, which is again typical for its behavior in most of the tests. In the 100K x 100 case, our tool performs the best, followed by Seqotron, which performs well with smaller files. For 1M x 100, our tool performs the best. While Seaview comes in second at initial value, Seqotron is better than that at peak value. Jalview is the worst. In the 1K X 1K test, our tool performs the best, followed by Seqotron. For 10K X 1K, our tool performs the best, followed by Seqotron. In 100K X 1K, Seaview performs slightly better than ours for initial value, but in terms of peak performance, our tool is the best. Aliview and Seqotron are better than Seaview for peak values. Jalview is the worst in both. For 100 X 10K, Seqotron performs the best, followed by our tool. In the 1K X 10K case, our tool performs the

best, followed by Seqotron. For 10K X 10K, our tool performs the best, followed by Seaview for initial value and Aliview for peak value. Jalview performs the worst. In 100K X 10K, Aliview performs the best, followed by our tool and then Seaview comes. The other two are much worse. For 10 X 100K, Seqotron performs the best, followed by our tool and then Seaview. There are no peak values because there are very few sequences, as scrolling is not available. In 100 X 100K, Seqotron performs the best, followed by Seaview and then our tool for initial values. In terms of peak values, our tool outperforms Seaview. For 1K X 100K, our tool performs the best, followed by Seaview for initial score and Aliview for peak score. Jalview again performs the worst. In 10K X 100K, our tool performs the best. It is closely followed by Seaview for initial value. Yet, Aliview outperform Seaview in regards to peak value. Jalview and Seqotron perform worse.

In summary, for 1MB and 10MB files, either our tool or Seqotron is the best, with one reason being that both apps consume less memory during startup (before file loading). For 100MB files, our tool is the best in all cases for peak values. For initial values, Seaview is the best in only one case, while ours is the best in others. While Seaview ranks second in initial values, Aliview outperforms it in peak scores. For 1GB files, our tool performs the best in one case, while Aliview leads in the other. In the case which we lead, Seaview comes second in respect to initial score but Aliview is better for peak score.

Based on our memory performance tests, our tool achieves better overall memory efficiency compared to existing alternatives. Notably, Aliview ranks second, as the peak memory score provides a more accurate performance measure than the initial score. This result is a significant achievement, given that competing tools have a long history of development and acceptance within the bioinformatics community.

4.3. Summary of Comparison

In summary, the proposed tool demonstrates significant advantages over existing alternatives such as Jalview, Seqotron, Seaview, and AliView. Key strengths include:

- Built on Biopython: Our platform is based on Biopython, ensuring rapid implementation, continuous updates, and extensive bioinformatics functionalities. Unlike other tools which rely on standalone implementations, our tool leverages Biopython's packages, offering greater flexibility and potential for future expansion. Additionally, many of the contributions and enhancements in our tool are made possible due to its foundation on Biopython.
- Wide File Format Support: Our tool supports 10/10 most common formats, surpassing alternatives like Jalview (4/10), Seaview (3/10), Aliview (3/10), and Seqotron (2/10). This extensive support ensures better compatibility across diverse datasets from many databases or software. Also, our tool can process files with multiple MSAs, unlike others, which have limitations in handling these files.
- Annotation Editor: Our platform fully supports annotations for annotation-rich formats (e.g. GenBank, EMBL, and UniProt-XML) and formats with few annotations with advanced features for editing and saving annotations. In comparison, other tools offer minimal or no annotation support.
- Extensive Pattern Search: Our tool provides advanced search capabilities (exact and PSSM-based matching) and pattern file handling, whereas other tools offer only basic search features.
- Index Mode: Handling Very Large Files: Our tool's efficient indexing process allows it to handle very large datasets seamlessly, without consuming excessive memory. This capability is only shared with Aliview.
- Memory Efficiency: Overall, our tool achieves the best memory performance, with Aliview ranking second. This is notable given the established history of competing tools.
- Cross-Platform Compatibility: Our tool, along with others like Jalview, Seaview, and Aliview, is cross-platform, while Seqotron is restricted to specific platforms.

Table 4.7 lists the upsides of our tool against others, summarizing the contributions of this work. As can be seen from the table, it can be stated that our tool is more advantageous than the others in many important criteria.

Table 4.7. Feature comparison table.

Feature	**BioPyView**	**Jalview**	**SeaView**	**AliView**	**Seqotron**
File Format Support	10/10	4/10	3/10	3/10	2/10
Reading Files with Multiple MSAs	Yes	-	-	-	-
Annotation Support	Full	Few	Few	None	None
Multiple pattern search	Yes	-	-	-	-
Approximate Search	Yes	-	-	-	-
Handling pattern files	Yes	-	-	-	-
Index Mode	Yes	-	-	Yes	-
Memory Performance	High	Low	Moderate	High	Low
Cross-Platform Compatibility	Yes	Yes	Yes	Yes	-

4.4. Limitations and Potential Improvements

Despite its many advantages, our tool requires improvements in render speed for long sequences, particularly in color mode. Our tests revealed a notable decrease in render performance when letters are colorized, especially for large sequences.

The tool supports letter-based coloring, but the rendering challenges stem from the working logic of the method offered by the GUI library used (Tkinter). Specifically, the Text widget employed for the sequence editor panel allows only a group of consecutive characters to be assigned a specific color at a time. Consequently, the coloring function must be invoked separately for each set of consecutive, identical characters, leading to a high number of function calls during each scroll event. This results in reduced render speed and a suboptimal user experience. To optimize performance, the tool colorizes only the visible region as the user scrolls through the file. As an alternative to this approach, pre-coloring the entire sequences at the beginning also does not resolve the issue, as scrolling through pre-colored content is also slow. Additionally, pre-coloring large files requires an infeasible amount of time during file loading.

Although other tools also experience scroll-related rendering issues for large files, it is evident that our tool requires more substantial improvements in render speed when using color mode. One potential solution involves managing the Text widget differently. Instead of inserting the entire file into the widget, only the visible area could be inserted and updated dynamically as the user navigates. This approach would allow for more efficient rendering and smoother scrolling. However, implementing this solution would necessitate developing custom implementations for most fundamental sequence viewer-editor functionalities, including scrolling, undo-redo operations, adding or deleting sequences, and tracking all changes occuring in the text widget. These fundamental features are typically handled automatically when inserting the entire file into the Text widget. However, by inserting only the visible area, full control shifts to us, requiring nearly a complete reimplementation of the Text widget's core functionalities.

An alternative solution could involve transitioning to other GUI libraries, such as PyQt, which may offer better performance for rendering tasks. However, adopting a different library would require significant modifications to the overall application structure as the entire GUI code, including windows, the text editor, and all event handling mechanisms, would need to be rewritten.

Addressing this limitation and exploring these potential improvements is critical for enhancing user experience and ensuring the tool's robustness for handling large sequences effectively.

5. CONCLUSION

The rapid advancements in sequencing technologies have led to an era of increasingly large and complex biological datasets. Analyzing DNA, RNA, and protein sequences has become integral to understanding the molecular mechanisms underpinning life, with multiple sequence alignments (MSAs) serving as the foundation for evolutionary, functional, and structural studies. However, despite the growth of bioinformatics tools, many suffer from limitations in functionality, usability, and scalability. This thesis has addressed these gaps by proposing and developing a modern, cross-platform, user-friendly sequence analysis tool built on Biopython.

5.1. Key Achievements

The proposed tool offers a robust platform for sequence analysis, combining essential features from existing software while introducing novel capabilities that set it apart from its counterparts. The integration of Biopython ensures support for a wide range of file formats, efficient parsing, and streamlined biological computations. In this work, we investigated four prominent counterparts: Jalview, Seaview, AliView, and Seqotron. The selection criteria for these tools were based on their publication in prestigious journals, their widespread use in the field, and the regular updates they receive. The following are the major contributions of this work:

5.1.1. Wide File Format Support

Our tool, built on Biopython, supports a significantly wider range of file formats compared to existing tools, including 38 formats versus 6 to 19 in other apps. It can handle both aligned and unaligned sequences, supporting all 10 of the most commonly used formats, unlike other tools that support far fewer. Additionally, our tool efficiently processes files containing multiple MSAs, overcoming the limitations observed in other tools which often fail to properly handle such datasets. This broad file format support

and the ability to manage complex datasets make our platform much more versatile and flexible, giving it a clear advantage over alternatives in bioinformatics analysis.

5.1.2. Advanced Annotation Support

Our platform offers advanced annotation support, fully handling key annotation-rich formats like GenBank, EMBL, and UniProt-XML, which contain detailed sequence and feature annotations. It includes an integrated viewer and editor with five specialized tabs for managing various annotation types, such as sequence details, feature tables, and per-letter annotations. In comparison, other tools like Jalview, Seaview, AliView, and Seqotron provide limited or no annotation support. The comprehensive annotation capabilities of our tool, including the ability to edit and save annotations, far exceed the offerings of other applications.

5.1.3. Enhanced Pattern Search

The proposed tool offers advanced pattern search capabilities, including both exact and PSSM-based approximate pattern matching, the ability to search specific regions, and support for opening and saving common pattern files. It also allows for multiple pattern searches, significantly enhancing its functionality for detailed sequence analysis. In contrast, existing tools offer very basic search features supporting only single pattern search but lacks the comprehensive functionalities of our tool, such as handling pattern files, supporting multiple pattern searches and approximate matching.

5.1.4. Index Mode: Handling Large Datasets

The proposed tool stands out for its ability to efficiently handle extremely large files through an indexing process, which indexes sequences and only caches them in memory when viewed. This method prevents memory overload and allows seamless processing of massive datasets. Unlike other tools our tool can manage enormous files exceeding RAM limits and this feature is absent in all alternatives except Aliview.

5.1.5. Memory Efficiency

We created mock DNA sequence files in FASTA format with varying sequence counts, lengths, and sizes, ranging from 1 MB to 1 GB, to evaluate the performance of our tool and compare it with other applications. The files mimicked real-world bioinformatics scenarios and testing was performed for both memory usage during startup and peak memory usage after interacting with the content. Overall, our tool outperformed others in memory efficiency and this achievement is significant given the longstanding development history of competing tools.

5.2. Limitations

Despite its advantages, our tool faces limitations in render speed for long sequences in color mode due to the performance constraints of the Tkinter Text widget, which requires frequent function calls for coloring consecutive characters. Addressing this may require dynamic insertion of visible areas or transitioning to alternative GUI libraries like PyQt, which could enhance performance but would involve significant restructuring.

5.3. Future Work

The proposed tool demonstrates significant potential; however, several improvements and extensions can be explored in the future to enhance its functionality and usability:

- Incorporating new Biopython components: Adding more Biopython functionalities can expand the tool's scope. Examples include:
  - Accessing key online databases: NCBI Entrez Utilities, ExPASy, InterPro, KEGG, and SCOP.
  - Handling 3D structures: Integration of PDB files for structural analysis.
  - BLAST search: Integration with both local and online BLAST databases.

  - Graphical output: Enhanced visualizations like Genome Diagrams and other graphical representations.

- Improving render speed in color mode: Enhancements are needed to address render speed issues for long sequences and in color mode, as discussed earlier.
- Enhancing editing capabilities in index mode: The current editing functionalities in index mode are more limited compared to normal mode. This is expected due to the nature of index mode, but there is still potential for further enhancement.
- Updating the UI for Windows OS: The default Tkinter UI appears a little outdated on Windows platforms. Modernizing the interface would improve the user experience significantly.
- Enhanced external tool support: More parameters for external alignment and tree tools could be displayed in the UI. Currently, only the most basic parameters can be adjusted.

5.4. Final Remarks

This thesis has demonstrated the feasibility and utility of building a modern sequence analysis platform on top of the Biopython library. By addressing key gaps in existing alternatives and leveraging the power of Biopython, this work offers a versatile and scalable platform that has many advantages over existing tools in several aspects. Considering that other tools have years of development history, have been developed by expert teams in the field, and have been published in well-regarded journals, the fact that we have developed a tool with such advantages in just one year, without a dedicated team effort, proves the success of this work. As sequencing technologies continue to advance, tools like the one proposed in this thesis will play an increasingly critical role in enabling biological discovery and innovation.

REFERENCES

1. Hall, T., "BioEdit: A Biological Sequence Alignment Editor", 1999, `https://thalljiscience.github.io`, accessed on January 5, 2025.

2. Cock, P., T. Antao, J. Chang, B. Chapman, C. Cox, A. Dalke, I. Friedberg, T. Hamelryck, F. Kauff, B. Wilczynski and M. de Hoon, "Biopython: Freely Available Python Tools for Computational Molecular Biology and Bioinformatics", *Bioinformatics*, Vol. 25, No. 11, pp. 1422–1423, 2009.

3. Waterhouse, A., J. Procter, D. Martin, M. Clamp and G. Barton, "Jalview Version 2–A Multiple Sequence Alignment Editor and Analysis Workbench", *Bioinformatics (Oxford, England)*, Vol. 25, No. 9, pp. 1189–1191, 2009.

4. Sievers, F., A. Wilm, D. Dineen, T. J. Gibson, K. Karplus, W. Li, R. Lopez, H. McWilliam, M. Remmert, J. Söding, J. D. Thompson and D. G. Higgins, "Fast, Scalable Generation of High-Quality Protein Multiple Sequence Alignments Using Clustal Omega", *Molecular Systems Biology*, Vol. 7, p. 539, 2011.

5. Edgar, R., "Muscle 5", 2021, `https://drive5.com/muscle5/`, accessed on January 5, 2025.

6. "FigTree Software", 2007, `http://tree.bio.ed.ac.uk/software/figtree/`, accessed on January 5, 2025.

7. Price, M., P. Dehal and A. Arkin, "FastTree 2–Approximately Maximum-Likelihood Trees for Large Alignments", *PloS One*, Vol. 5, No. 3, p. e9490, 2010.

8. Stamatakis, A., "RAxML Version 8: A Tool for Phylogenetic Analysis and Post-Analysis of Large Phylogenies", *Bioinformatics*, Vol. 30, No. 9, pp. 1312–1313, 2014.

9. Gouy, M., S. Guindon and O. Gascuel, "SeaView Version 4: A Multiplatform Graphical User Interface for Sequence Alignment and Phylogenetic Tree Building", *Molecular Biology and Evolution*, Vol. 27, No. 2, pp. 221–224, 2010.

10. Larsson, A., "AliView: A Fast and Lightweight Alignment Viewer and Editor for Large Datasets", *Bioinformatics (Oxford, England)*, Vol. 30, No. 22, pp. 3276–3278, 2014.

11. Fourment, M. and E. C. Holmes, "Seqotron: A User-Friendly Sequence Editor for Mac OS X", *BMC Research Notes*, Vol. 9, p. 106, 2016.

12. "Jalview Software", 2003, `https://www.jalview.org`, accessed on January 5, 2025.

13. "SeaView Software", 2014, `https://doua.prabi.fr/software/seaview3`, accessed on January 5, 2025.

14. "AliView", 2014, `https://ormbunkar.se/aliview/`, accessed on January 5, 2025.

15. "Seqotron Software", 2015, `https://github.com/4ment/seqotron`, accessed on January 5, 2025.

16. "Jalview 2.11 Manual and Introductory Tutorial", School of Life Sciences, University of Dundee, Dundee, Scotland DD1 5EH, UK, 2020.

17. "Seaview 5.0.5 On-line Help Document", 2004, `https://doua.prabi.fr/software/seaview`, accessed on January 5, 2025.

18. "Seqotron 1.0 User Manual", University of Technology Sydney, 2015.

19. Thompson, J. D., D. G. Higgins and T. J. Gibson, "CLUSTAL W: Improving the Sensitivity of Progressive Multiple Sequence Alignment Through Sequence Weighting, Position-Specific Gap Penalties and Weight Matrix Choice", *Nucleic Acids*

Research, Vol. 22, No. 22, pp. 4673–4680, 1994.

20. Katoh, K., K. Kuma, H. Toh and T. Miyata, "MAFFT Version 5: Improvement in Accuracy of Multiple Sequence Alignment", *Nucleic Acids Research*, Vol. 33, No. 2, pp. 511–518, 2005.

21. Do, C. B., M. S. Mahabhashyam, M. Brudno and S. Batzoglou, "ProbCons: Probabilistic Consistency-Based Multiple Sequence Alignment", *Genome Research*, Vol. 15, No. 2, pp. 330–340, 2005.

22. Notredame, C., D. G. Higgins and J. Heringa, "T-Coffee: A Novel Method for Fast and Accurate Multiple Sequence Alignment", *Journal of Molecular Biology*, Vol. 302, No. 1, pp. 205–217, 2000.

23. Guindon, S. and O. Gascuel, "A Simple, Fast, and Accurate Algorithm to Estimate Large Phylogenies by Maximum Likelihood", *Systematic Biology*, Vol. 52, No. 5, pp. 696–704, 2003.

24. Fourment, M. and E. Holmes, "Novel Non-Parametric Models to Estimate Evolutionary Rates and Divergence Times from Heterochronous Sequence Data", *BMC Evolutionary Biology*, Vol. 14, p. 163, 2014.

25. Cole, C., J. Barber and G. Barton, "The JPred 3 Secondary Structure Prediction Server", *Nucleic Acids Research*, Vol. 36, No. Web Server issue, pp. W197–W201, 2008.

26. "BioPyView: Biopython-based Platform for Sequence Analysis", 2025, `https://github.com/omerfarukcavass/biopyview`, accessed on January 5, 2025.

27. Lafita, A., S. Bliven, A. Prlić, D. Guzenko, P. W. Rose, A. Bradley, P. Pavan, D. Myers-Turnbull, Y. Valasatava, M. Heuer, M. Larson, S. K. Burley and J. M. Duarte, "BioJava 5: A Community Driven Open-Source Bioinformatics Library",

PLoS Computational Biology, Vol. 15, No. 2, p. e1006791, 2019.

28. Stajich, J. E., D. Block, K. Boulez, S. E. Brenner, S. A. Chervitz, C. Dagdigian, G. Fuellen, J. G. R. Gilbert, I. Korf, H. Lapp, H. Lehväslaiho, C. Matsalla, C. J. Mungall, B. I. Osborne, M. R. Pocock, P. Schattner, M. Senger, L. D. Stein, E. Stupka, M. D. Wilkinson and E. Birney, "The BioPerl Toolkit: Perl Modules for the Life Sciences", *Genome Research*, Vol. 12, No. 10, pp. 1611–1618, 2002.

29. "PyQt Documentation (Riverbank Computing)", 2008, `https://www.riverbankcomputing.com/software/pyqt/intro`, accessed on January 5, 2025.

30. "Memory Profiler", 2011, `https://pypi.org/project/memory-profiler/`, accessed on January 5, 2025.

31. "Pympler 1.1 Documentation", 2008, `https://pympler.readthedocs.io/`, accessed on January 5, 2025.

32. "PyInstaller Manual", 2005, `https://pyinstaller.org/`, accessed on January 5, 2025.

33. "Biopython SeqIO", 2000, `https://biopython.org/wiki/SeqIO`, accessed on January 5, 2025.

34. "Biopython AlignIO", 2000, `https://biopython.org/wiki/AlignIO`, accessed on January 5, 2025.

35. Löytynoja, A., "PRANK", 2010, `https://ariloytynoja.github.io/prank-msa/`, accessed on January 5, 2025.

36. Liu, Y., B. Schmidt and D. L. Maskell, "MSAProbs: Multiple Sequence Alignment Based on Pair Hidden Markov Models and Partition Function Posterior Probabilities", *Bioinformatics (Oxford, England)*, Vol. 26, No. 16, pp. 1958–1964, 2010.

37. Rice, P., I. Longden and A. Bleasby, "EMBOSS: The European Molecular Biology Open Software Suite", *Trends in Genetics: TIG*, Vol. 16, No. 6, pp. 276–277, 2000.

38. "Biopython Tutorial", 2000, `https://biopython.org/docs/latest/Tutorial/`, accessed on January 5, 2025.

39. "Tkinter Documentation (Python Software Foundation)", 2001, `https://docs.python.org/3/library/tkinter.html`, accessed on January 5, 2025.

APPENDIX A: ANNOTATION WINDOWS

This appendix includes several figures from annotation component of the application.

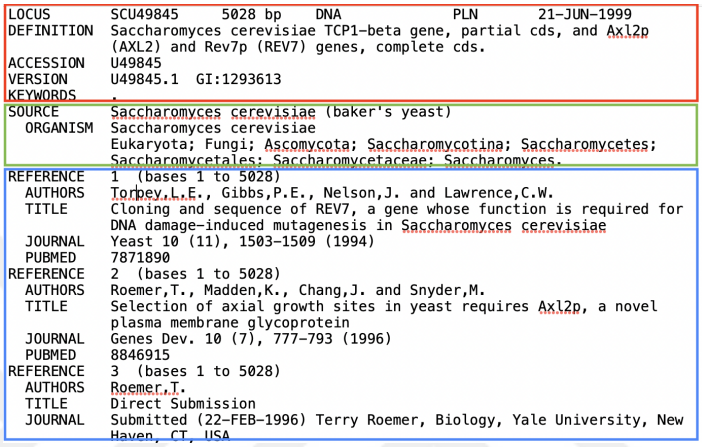


Figure A.1. The annotations in the GenBank file.

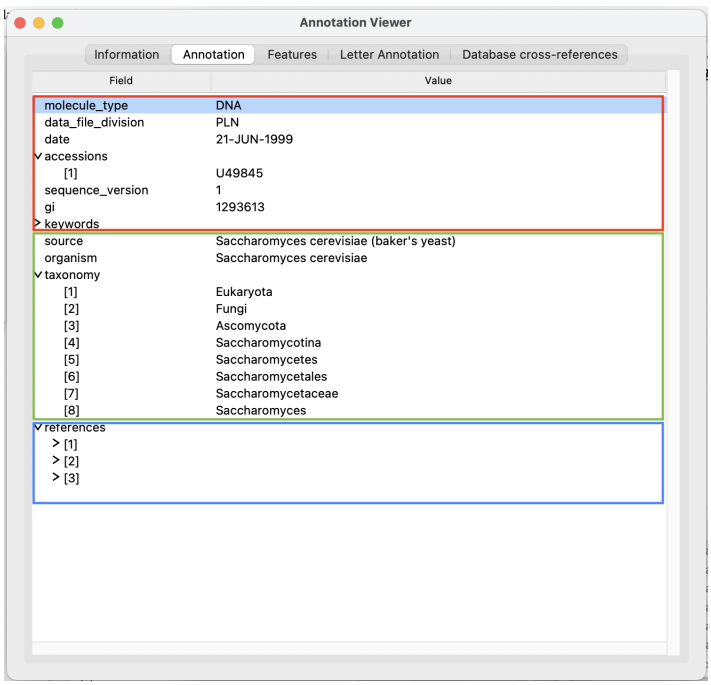


Figure A.2. The annotations displayed in the app's annotation tab.

```
REFERENCE   1  (bases 1 to 5028)
  AUTHORS   Torpey,L.E., Gibbs,P.E., Nelson,J. and Lawrence,C.W.
  TITLE     Cloning and sequence of REV7, a gene whose function is required for
            DNA damage-induced mutagenesis in Saccharomyces cerevisiae
  JOURNAL   Yeast 10 (11), 1503-1509 (1994)
  PUBMED    7871890
REFERENCE   2  (bases 1 to 5028)
  AUTHORS   Roemer,T., Madden,K., Chang,J. and Snyder,M.
  TITLE     Selection of axial growth sites in yeast requires Axl2p, a novel
            plasma membrane glycoprotein
  JOURNAL   Genes Dev. 10 (7), 777-793 (1996)
  PUBMED    8846915
REFERENCE   3  (bases 1 to 5028)
  AUTHORS   Roemer,T.
  TITLE     Direct Submission
  JOURNAL   Submitted (22-FEB-1996) Terry Roemer, Biology, Yale University, New
            Haven, CT, USA
```

Figure A.3. Reference annotations in the GenBank file.

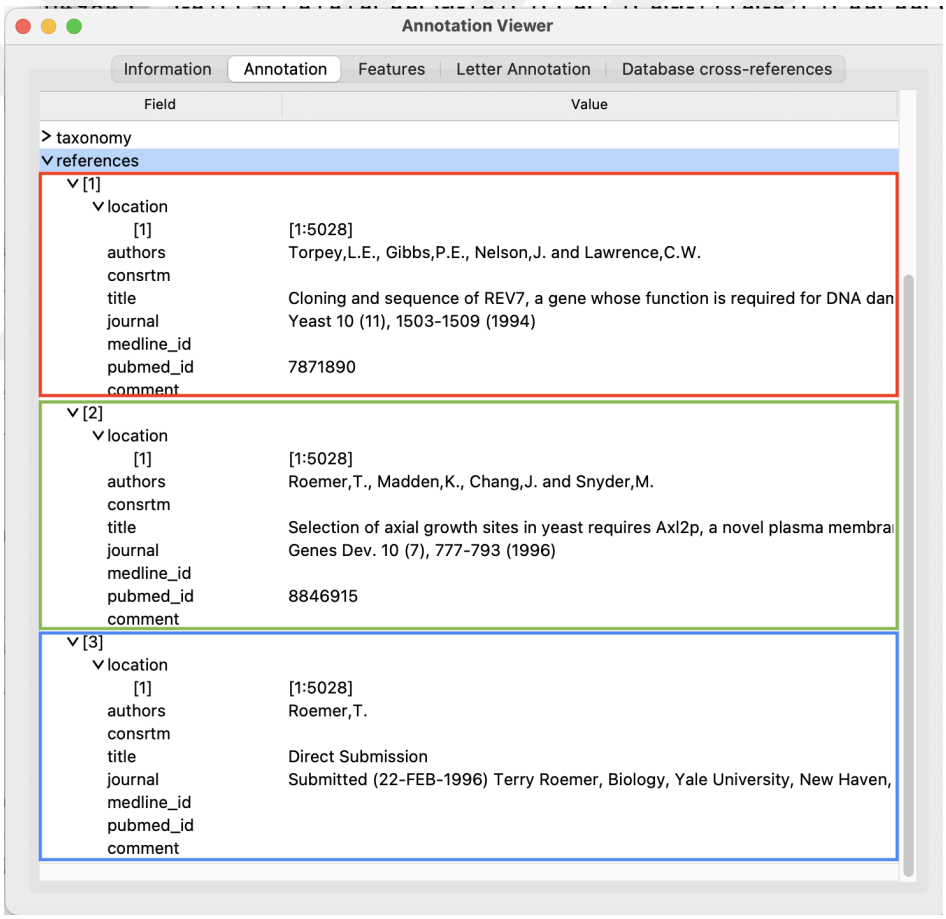


Figure A.4. Reference annotations in the app's annotation tab.

```
FEATURES             Location/Qualifiers
     source          1..5028
                     /organism="Saccharomyces cerevisiae"
                     /db_xref="taxon:4932"
                     /chromosome="IX"
                     /map="9"
     CDS             <1..206
                     /codon_start=3
                     /product="TCP1-beta"
                     /protein_id="AAA98665.1"
                     /db_xref="GI:1293614"
                     /translation="SSIYNGISTSGLDLNNGTIADMRQLGIVESYKLKRAVVSSASEA
                     AEVLLRVDNIIRARPRTANRQHM"
     gene            687..3158
                     /gene="AXL2"
     CDS             687..3158
                     /gene="AXL2"
                     /note="plasma membrane glycoprotein"
                     /codon_start=1
                     /function="required for axial budding pattern of S.
                     cerevisiae"
                     /product="Axl2p"
                     /protein_id="AAA98666.1"
                     /db_xref="GI:1293615"
                     /translation="MTQLQISLLLTATISLLHLVVATPYEAYPIGKQYPPVARVNESF
                     TFQISNDTYKSSVDKTAQITYNCFDLPSWLSFDSSSRTFSGEPSSDLLSDANTTLYFN
                     VILEGTDSADSTSLNNTYQFVVTNRPSISLSSDFNLLALLKNYGYTNGKNALKLDPNE
                     VFNVTFDRSMFTNEESIVSYYGRSQLYNAPLPNWLFFDSGELKFTGTAPVINSAIAPE
                     TSYSFVIIATDIEGFSAVEVEFELVIGAHQLTTSIQNSLIINVTDTGNVSYDLPLNYV
                     YLDDDPISSDKLGSINLLDAPDWVALDNATISGSVPDELLGKNSNPANFSVSIYDTYG
                     DVIYFNFEVVSTTDLFAISSLPNINATRGEWFSYYFLPSQFTDYVNTNVSLEFTNSSQ
                     DHDWVKFQSSNLTLAGEVPKNFDKLSLGLKANQGSQSQELYFNIIGMDSKITHSNHSA
                     NATSTRSSHHSTSTSSYTSSTYTAKISSTSAAATSSAPAALPAANKTSSHNKKAVAIA
                     CGVAIPLGVILVALICFLIFWRRRRENPDDENLPHAISGPDLNNPANKPNQENATPLN
                     NPFDDDASSYDDTSIARRLAALNTLKLDNHSATESDISSVDEKRDSLSGMNTYNDQFQ
                     SQSKEELLAKPPVQPPESPFFDPQNRSSSVYMDSEPAVNKSWRYTGNLSPVSDIVRDS
                     YGSQKTVDTEKLFDLEAPEKEKRTSRDVTMSSLDPWNSNISPSPVRKSVTPSPYNVTK
                     HRNRHLQNIQDSQSGKNGITPTTMSTSSSDDFVPVKDGENFCWVHSMEPDRRPSKKRL
                     VDFSNKSNVNVGQVKDTHGRIPEML"
     gene            complement(3300..4037)
                     /gene="REV7"
     CDS             complement(3300..4037)
                     /gene="REV7"
                     /codon_start=1
                     /product="Rev7p"
                     /protein_id="AAA98667.1"
                     /db_xref="GI:1293616"
                     /translation="MNRWVEKWLRVYLKCYINLILFYRNVYPPQSFDYTTYQSFNLPQ
                     FVPINRHPALIDYIEELILDVLSKLTHVYRFSICIINKKNDLCIEKYVLDFSELQHVD
                     KDDQIITETEVFDEFRSSLNSLIMHLEKLPKVNDDTITFEAVINAIELELGHKLDRNR
                     RVDSLEEKAEIERDSNWVKCQEDENLPDNNGFQPPKIKLTSLVGSDVGPLIIHQFSEK
                     LISGDDKILNGVYSQYEEGESIFGSLF"
```

Figure A.5. Feature table elements in the GenBank file.

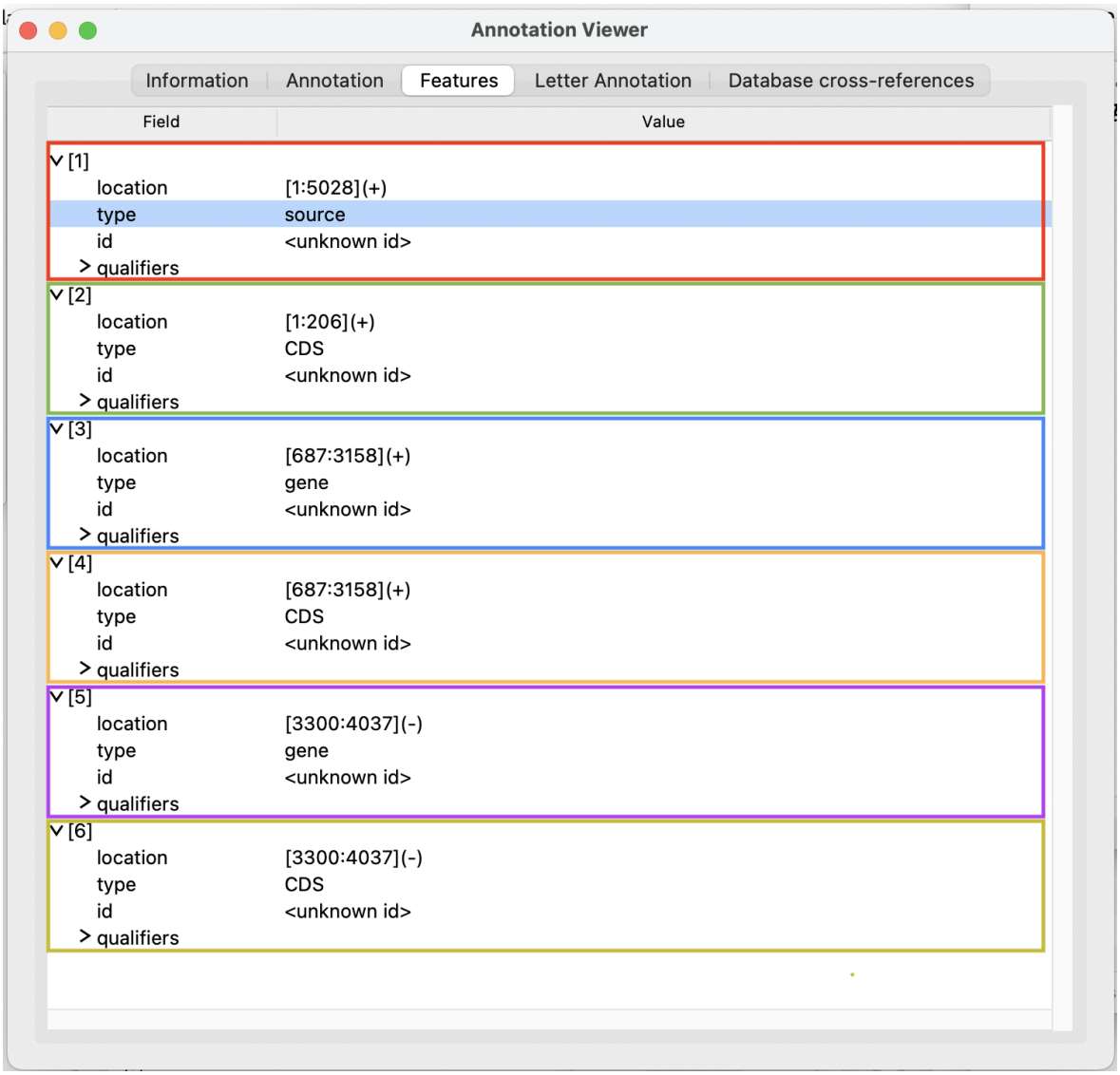


Figure A.6. Feature table elements in the app.

```
CDS             687..3158
                /gene="AXL2"
                /note="plasma membrane glycoprotein"
                /codon_start=1
                /function="required for axial budding pattern of S.
                cerevisiae"
                /product="Axl2p"
                /protein_id="AAA98666.1"
                /db_xref="GI:1293615"
                /translation="MTQLQISLLLTATISLLHLVVATPYEAYPIGKQYPPVARVNESF
                TFQISNDTYKSSVDKTAQITYNCFDLPSWLSFDSSSRTFSGEPSSDLLSDANTTLYFN
                VILEGTDSADSTSLNNTYQFVVTNRPSISLSSDFNLLALLKNYGYTNGKNALKLDPNE
                VFNVTFDRSMFTNEESIVSYYGRSQLYNAPLPNWLFFDSGELKFTGTAPVINSAIAPE
                TSYSFVIIATDIEGFSAVEVEFELVIGAHQLTTSIQNSLIINVTDTGNVSYDLPLNYV
                YLDDDPISSDKLGSINLLDAPDWVALDNATISGSVPDELLGKNSNPANFSVSIYDTYG
                DVIYFNFEVVSTTDLFAISSLPNINATRGEWFSYYFLPSQFTDYVNTNVSLEFTNSSQ
                DHDWVKFQSSNLTLAGEVPKNFDKLSLGLKANQGSQSQELYFNIIGMDSKITHSNHSA
                NATSTRSSHHSTSTSSYTSSTYTAKISSTSAAATSSAPAALPAANKTSSHNKKAVAIA
                CGVAIPLGVILVALICFLIFWRRRRENPDDENLPHAISGPDLNNPANKPNQENATPLN
                NPFDDDASSYDDTSIARRLAALNTLKLDNHSATESDISSVDEKRDSLSGMNTYNDQFQ
                SQSKEELLAKPPVQPPESPFFDPQNRSSSVYMDSEPAVNKSWRYTGNLSPVSDIVRDS
                YGSQKTVDTEKLFDLEAPEKEKRTSRDVTMSSLDPWNSNISPSPVRKSVTPSPYNVTK
                HRNRHLQNIQDSQSGKNGITPTTMSTSSSDDFVPVKDGENFCWVHSMEPDRRPSKKRL
                VDFSNKSNVNVGQVKDIHGRIPEML"
```

Figure A.7. CDS feature in the GenBank file.

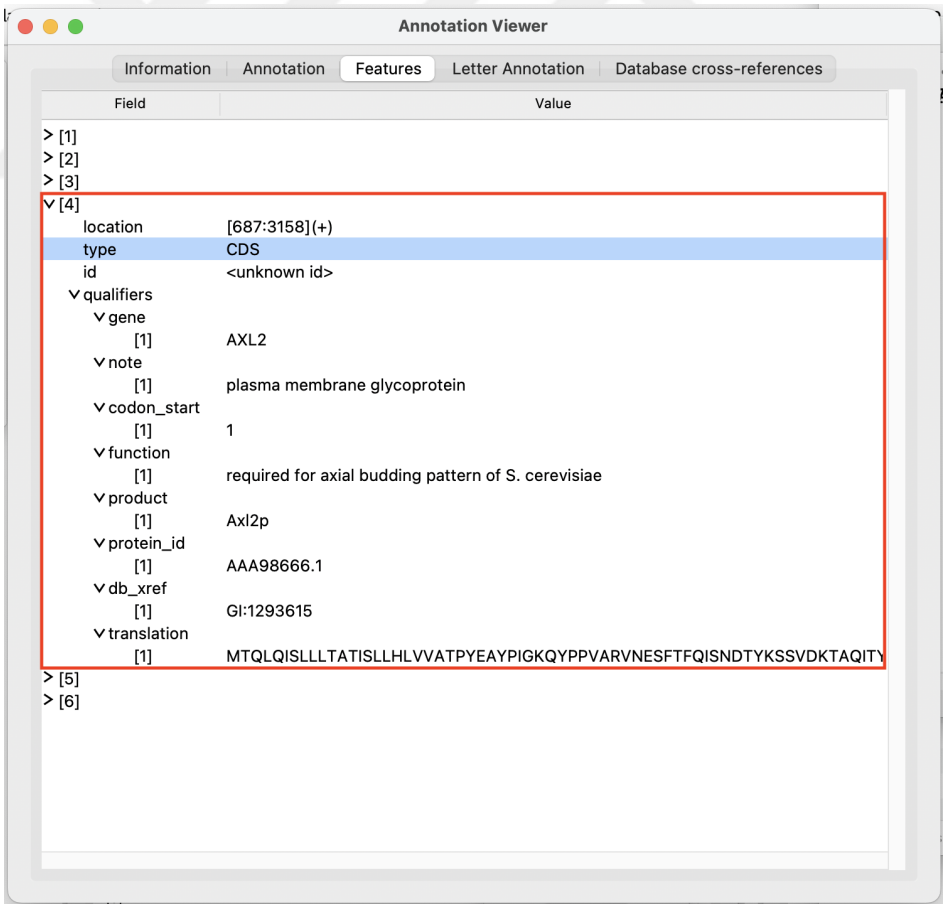


Figure A.8. CDS feature in the app.