MolecularBio: A Python Toolkit for Molecular Biology Data Processing

Rafail Konstantinou inf2022248
Dimitris Blougouras inf2022134
Pangiotis Stoupis inf2022199

May 26, 2025

Contents

1 Introduction

MolecularBio is a Python-based command-line application designed to simplify the processing and analysis of molecular biology data. The toolkit allows users to parse structured input files, perform statistical analyses, and generate visualizations from biological datasets.

The project is ideal for bioinformatics learners and researchers seeking a lightweight tool for analyzing DNA/RNA sequences and protein data in formats like FASTA and CSV.

2 Project Objectives

The primary objectives of MolecularBio are:

- To simplify molecular biology data processing through an intuitive Python interface.
- To provide statistical analysis functions such as sequence length distribution, nucleotide frequency, and motif search.
- To create visualizations for better understanding of gene/protein data.
- To be easily deployable through Docker and platform-independent.

3 Installation

3.1 Using pip

```
$ git clone
$ cd molecularbio
$ pip install -r requirements.txt
$ python app.py
```

3.2 Using Docker

```
$ docker build -t molecularbio .
$ docker run -p 8000:8000 molecularbio
```

4 Directory Structure

```
molecularbio/
 data/                  # Sample FASTA/CSV files
 app.py                 # CLI interface
 requirements.txt       # Python dependencies
 Dockerfile             # Docker configuration
 README.md
```

5 Code Breakdown

5.1 app.py

Responsible for parsing FASTA and CSV files.

Listing 1: Sample FASTA Parsing Code

```
def parse_fasta(file_path):
    with open(file_path, 'r') as file:
        sequences = {}
        sequence = ''
        header = ''
        for line in file:
            if line.startswith('>'):
                if header:
                    sequences[header] = sequence
                header = line.strip()[1:]
                sequence = ''
            else:
                sequence += line.strip()
        sequences[header] = sequence
    return sequences
```

5.2 app.py

Provides core statistical functionality.

Listing 2: Basic Nucleotide Frequency

```
from collections import Counter

def nucleotide_frequency(sequence):
    return Counter(sequence.upper())
```

5.3 app.py

Uses matplotlib to visualize distributions.

Listing 3: Plotting Sequence Lengths

```
import matplotlib.pyplot as plt

def plot_sequence_lengths(sequences):
    lengths = [len(seq) for seq in sequences.values()]
    plt.hist(lengths, bins=20, color='skyblue', edgecolor='black')
    plt.title("Sequence Length Distribution")
    plt.xlabel("Length")
    plt.ylabel("Frequency")
    plt.show()
```

5.4 app.py

Main application that connects all components and parses CLI arguments.

Listing 4: Main Entry Point

```

if __name__ == "__main__":
    parser = argparse.ArgumentParser(description="MolecularBio CLI")
    parser.add_argument("--input", type=str, required=True)
    parser.add_argument("--analyze", action="store_true")
    parser.add_argument("--visualize", action="store_true")
    args = parser.parse_args()

    data = parser.parse_fasta(args.input)

    if args.analyze:
        for header, seq in data.items():
            print(header, analysis.nucleotide_frequency(seq))

    if args.visualize:
        visualize.plot_sequence_lengths(data)
```

6 Dockerization

To containerize the app:

Dockerfile example:

Listing 5: Dockerfile

```
FROM python:3.10
WORKDIR /app
COPY . .
RUN pip install -r requirements.txt
CMD ["python", "app.py", "--input", "data/sample.fasta", "--visualize"]
```

7 Future Improvements

- Integration with BioPython for more advanced sequence manipulation.
- Web-based interface using Flask or FastAPI.
- Support for multiple sequence alignments.
- More statistical models and pattern discovery tools.

8 Conclusion

MolecularBio serves as a lightweight and modular tool for basic molecular biology data analysis. It is intended as an educational and practical tool for those new to bioinformatics and Python scripting in the life sciences.

For source code, updates, and collaboration, visit the GitHub repository at: