DNA-Sequence-Analyzer

Course: INFO B 573 Programming Science for Bioinformatics

Prasanth Kumar Thuthika
MS Bioinformatics

Background

Bioinformatics is a rapidly growing field at the intersection of biology and computer science. With the vast amounts of DNA sequence data generated through modern sequencing technologies, computational tools are essential for analyzing, interpreting, and deriving meaningful insights.

This project, DNA Sequence Analyzer, was my first experience building a Python-based tool. Coming from a wet lab background and first hands on coding, I developed this project to learn Python programming and explore its applications in bioinformatics.

The tool was created to analyze DNA sequences for basic attributes like nucleotide composition, GC content, and to perform simple bioinformatics tasks like translating DNA into protein sequences and identifying restriction enzyme cut sites. This project was a learning exercise to understand how bioinformatics workflows can be implemented programmatically.

Introduction

The DNA Sequence Analyzer was designed to simplify the analysis of DNA sequences. It accepts user input either as a raw sequence or from a FASTA file and performs several analyses:

- **Nucleotide Composition**: Calculates GC content and AT content.
- **Translation**: Converts the DNA sequence into a protein sequence using the standard genetic code.
- **Restriction Enzyme Analysis**: Identifies cut sites for common restriction enzymes, including EcoRI, BamHI, and HindIII.
- **Mutation Simulation**: Simulates random mutations, including substitutions, insertions and deletions based on user-defined probabilities.
- **Codon Usage**: Analyzes the frequency of codons to identify potential biases in the DNA sequence.
- **Output Options**: Displays results in the terminal and provides an option to save them to a .txt file for future reference.

Python Script – Dependencies and Libraries

The project is implemented in Python, leveraging key libraries for bioinformatics and data processing.

Dependencies

- **Python Version**: Python 3.12.
- **Libraries**:
  - **Biopython**: For sequence translation and GC content calculation [1].
  - **tqdm**: For displaying progress bars during restriction enzyme analysis.
  - **collections.Counter**: For calculating nucleotide and codon frequencies.
  - **random**: For simulating mutations in the DNA sequence.
  - **os**: For file handling, particularly for reading FASTA files.

Code Modularity

- analysis.py: Contains all reusable functions for DNA sequence analysis.
- main.py: Handles user interaction and integrates functions from analysis.py.

Run the script

Use the command to run the tool```python main.py```

Results and Usage

The DNA Sequence Analyzer provides the option for the users to input sequences either directly or through a FASTA file. Below is an example output for a sample sequence:

Input Sequence:
ATTAGGACCAATATTATTAAGGACCATATATAGACACATATATAGGACACATTATATATTAAAGAGGAACCACACACATATTA

Output:

```
Welcome to the DNA Sequence Analyzer!
Enter your DNA sequence:
ATTAGGACCAATATTATTAAGGACCATATATAGACACATATATAGGACACATTATATATTAAAGAGGAACCACACACATATTA
Sequence is valid! Length: 83

--- Nucleotide Content ---
GC Content: 27.71%
AT Content: 72.29%

--- Nucleotide Frequencies ---
A: 38
T: 22
C: 13
G: 10

--- Translated Protein Sequence ---
IRTNIIKDHI*THI*DTLYIKEEPHTY

--- Restriction Enzyme Sites ---
No restriction sites found.

--- Codon Usage ---
ATT: 4
AGG: 1
ACC: 1
AAT: 1
AAG: 1
GAC: 2
CAT: 2
ATA: 2
TAG: 2
ACA: 3
TTA: 1
TAT: 2
AAA: 1
GAG: 1
GAA: 1
CCA: 1
CAC: 1

--- Mutated DNA Sequence ---
AGTAGGACCAACGCTATTCGGCACGCACTACGCACATGAGGAACCACACACCATA

--- Mutated Protein Sequence ---
SRTTALIARTTTR
```

Significance of the Project

This project served as a practical exercise to:

1. Learn the basics of Python programming.
2. Implement modular coding practices.
3. Understand how to work with external libraries such as Biopython.
4. Gain confidence in writing scripts for basic bioinformatics tasks.

While the tool is simple, it covers foundational programming and bioinformatics concepts that can be extended or enhanced further.

Conclusion

The **DNA Sequence Analyzer** was a valuable learning experience, helping me understand how to translate basic bioinformatics workflows into Python code. It also allowed me to explore how programming can simplify tasks often performed manually in the lab. This project has given me a good starting point for more advanced projects in bioinformatics.

References

1. Cock, P.J.A., et al. (2009). Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25(11), 1422–1423.

2. tqdm Documentation: https://github.com/tqdm/tqdm

3. Python Official Documentation: https://docs.python.org/3/