Department of Artificial Intelligence

22BIO201: Intelligence of Biological System - I

NOV-2024

Project Report

Chaos Game Representation of DNA and Proteins

Team Members:

Diya Deepak (CB.SC.U4AIE23020)

Ghanasree S (CB.SC.U4AIE23028)

Neha Jacob (CB.SC.U4AIE23046)

Sriranjana.C (CB.SC.U4AIE23066)

Date of submission: 08/11/2024

Signature of the Project Supervisor:

Abstract

This project explores the use of Chaos Game Representation (CGR) for the visualization and analysis of DNA and protein sequences, providing a unique approach to revealing patterns and motifs within genetic data. CGR is a fractal-based method that transforms sequences into high-dimensional graphical representations, where each base or amino acid is assigned specific coordinates. By iteratively mapping each element in the sequence, CGR generates visually distinctive patterns that reflect the structure and composition of the underlying data. This project implements a web-based tool using the Flask framework, enabling users to upload sequence files in FASTA format and generate CGR images for both DNA and protein sequences.

For DNA, each nucleotide is mapped within a square, while for proteins, amino acids can be plotted using a dodecahedral layout, providing detailed visualizations of sequence composition. The resulting CGR plots can highlight structural motifs, repetitive sequences, and evolutionary features, making this tool a valuable asset in genomics research. Overall, this project demonstrates CGR's potential to enhance bioinformatics analysis by offering an accessible, efficient means of visualizing and interpreting complex biological sequences.

1) Introduction

1.1) Background of Chaos Game Representation

Chaos Game Representation (CGR) is a graphical technique originally developed as a fractal-based approach to visualize sequences of symbols in a two-dimensional space. This method, used by Jeffrey in 1990, has since gained popularity for visualizing biological sequences, such as DNA and proteins, due to its ability to encode sequence data into unique, visually interpretable patterns. The term "chaos" in CGR refers to the process of iteratively mapping symbols (like nucleotide bases or amino acids) within a geometric space, producing complex patterns that appear chaotic yet are deterministic and reproducible. In CGR, each symbol is assigned specific coordinates, allowing sequences of any length to be transformed into intricate fractal images.

In biological contexts, CGR provides a method for transforming DNA and protein sequences into two-dimensional or three-dimensional patterns, capturing both the composition and the order of sequence elements. Unlike traditional alignment techniques, which rely on base pair comparisons, CGR is alignment-free and computationally efficient, making it ideal for large-scale genomic studies. By examining CGR images, we can identify structural motifs, repetitive patterns, and anomalies within sequences, which may reveal important biological information about gene function, evolutionary relationships, and genetic variability.

1.2) Importance of CGR in bioinformatics

By transforming linear sequences into distinct patterns, CGR enables researchers to quickly visualize sequence motifs, repetitive elements, and compositional trends, aiding in comparative genomics and evolutionary studies. Through the condensed, visually accessible format of CGR, large genomic datasets can be represented compactly, making it easier to detect sequence similarities and anomalies, which are crucial for understanding genomic structure and evolution.

Because of its properties, CGR has already been used, for instance, in alignment-free sequence comparison, phylogeny, and as an encoding for machine learning, and has also a huge potential for future applications in bioinformatics.

1.3) Objective and Scope of the project

The primary objective of this project is to create a web-based tool using CGR to visualize DNA and protein sequences. By implementing a graphical user interface (GUI) using Flask, this project enables users to upload biological sequence files in FASTA format and generate CGR plots, allowing them to analyze sequence patterns with minimal computational effort. The tool includes options to generate CGR images for both DNA and protein sequences.

In DNA CGR, each nucleotide—adenine (A), cytosine (C), guanine (G), and thymine (T)—is mapped to the corners of a square. As the sequence is processed, each nucleotide directs the plot point toward its corresponding corner, producing a fractal pattern that reflects the sequence's composition and structure. For protein CGR, the complexity increases as each of the 20 amino acids must be mapped to specific coordinates within a geometric structure, such as a dodecahedron. This mapping helps capture the intricacies of protein sequences, which have a more diverse set of elements than DNA sequences.

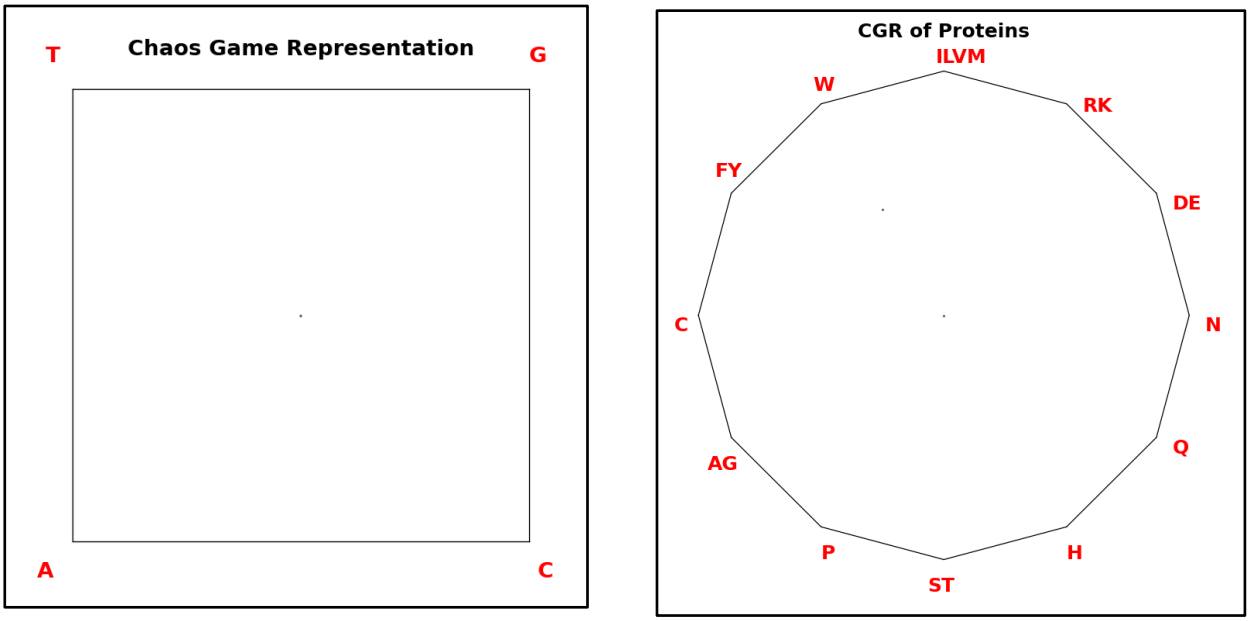


Fig 1 - Chaos Game Representation of DNA (left) and Proteins (right)

1.4) Relevance of DNA and Protein visualization

Understanding DNA and protein sequences is essential for research in genetics, molecular biology, and evolutionary studies. DNA, the molecule that carries genetic instructions, consists of long sequences of nucleotides that encode genes, which are responsible for hereditary traits and biological functions. The visualization of DNA sequences through CGR can reveal structural motifs and regulatory elements, providing insight into gene organization and potential areas of biological interest

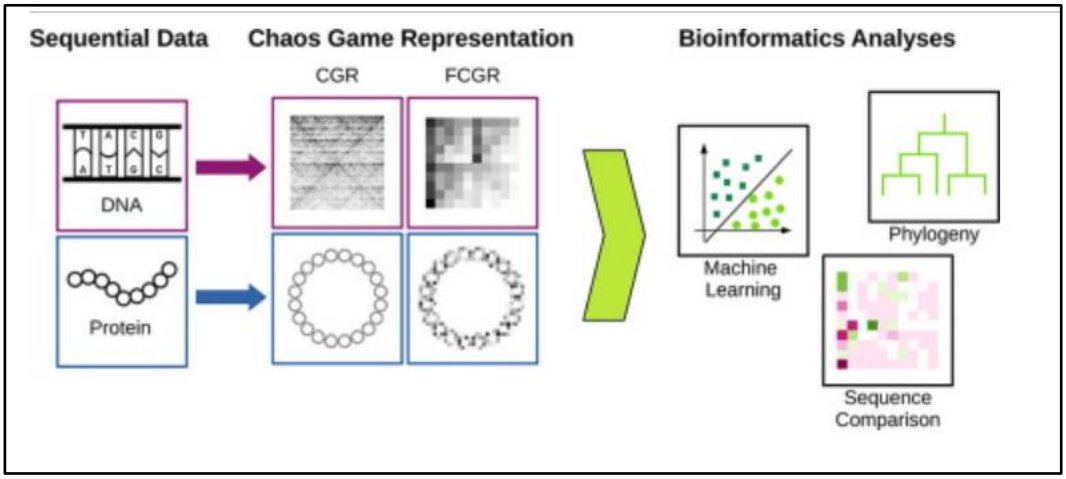


Fig 2 - Relevance of DNA and Protein visualisation

Protein sequences, composed of chains of amino acids, determine the structure and function of proteins, which perform critical roles in cellular processes. Each protein's sequence specifies its three-dimensional structure, influencing its stability, activity, and interactions within the cell. By visualizing protein sequences with CGR, we can identify patterns in amino acid distribution, highlight conserved regions, and uncover relationships between protein families. This information is valuable for applications in drug design, disease research, and biotechnology, where understanding protein structure-function relationships is crucial.

Team Contribution –

- Ghansree – Literature Review and Background research , Dataset collection on CGR
- Sriranjana – Developing and testing the CGR algorithm and Doucmentaion
- Neha Jacob – Design and Implementation of the web based interface for the CGR tool
- Diya Deepak – Testing and validation of the tool and Dataset collection

2) Related work

2.1) History of CGR

The chaos game algorithm has been developed by Barnsley to construct fractals based on random input and was later extended to DNA as input by Jeffrey. Fractals are recursive, scale-invariant patterns and their dimension is a fraction. The underlining concept of the CGR algorithm is to map a sequence, i.e., a 1D representation to a higher dimensional space, typically to the 2D space.

It was originally developed to construct the Sierpinski triangle. To this end, numbers from one to three are assigned on the vertices of a triangle. Based on a randomly selected start point (S), a vertex is randomly chosen (V1), and a point P1 is drawn in half the distance to the vertex V1. This process is repeated, with P1 as the new starting point. The second point (P2) is drawn at half the way to the second randomly selected vertex (V2). By repeating this algorithm, the Sierpinski triangle emerges.

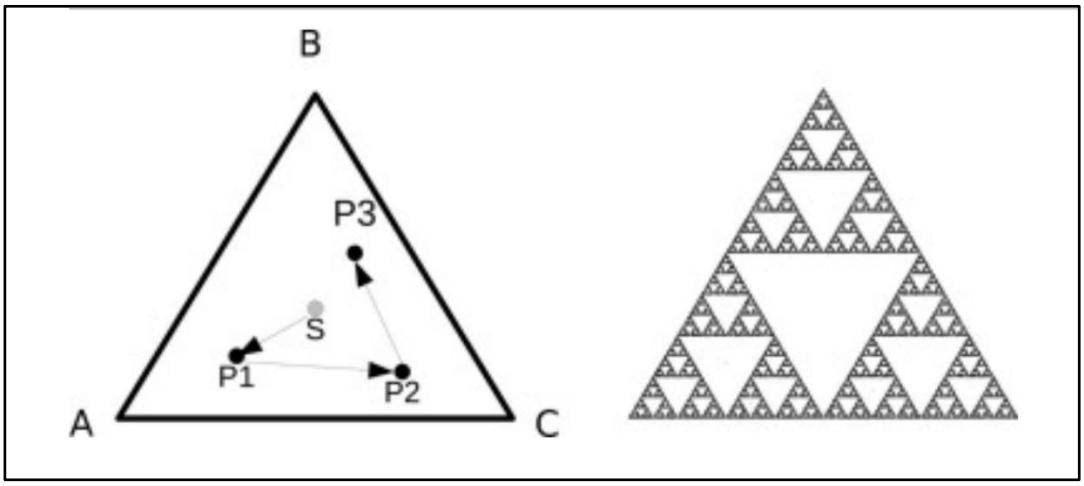


Fig 3 - Sierpinski triangle

This concept was later experimented with regular pentagons and hexagons which produced interesting fractal flakes. A fractal pattern is a detailed, self-repeating geometric shape that appears similar at various scales, meaning each smaller part is a miniature version of the whole. These patterns are seen in nature, such as in snowflakes and tree branching, where complex shapes repeat endlessly in a smaller form.

2.2) DNA and the need for CGR

DNA (Deoxyribonucleic Acid) is the hereditary material in all living organisms, carrying the genetic instructions used in the growth, development, and functioning of cells. It is composed of two long chains of nucleotides twisted into a double helix, with each nucleotide containing a sugar, phosphate group, and a nitrogenous base (adenine, thymine, cytosine, and guanine). DNA sequences encode the information necessary to produce proteins, which perform most cellular functions. DNA is crucial as it stores the genetic blueprint of an organism and is responsible for passing traits from one generation to the next. CGR was quickly recognized as a powerful visualization tool, enabling researchers to analyse genetic sequences in a novel way that bypasses traditional alignment techniques. More recent studies have explored CGR's application in large-scale genome comparisons, showing that CGR patterns can serve as "genomic fingerprints," where each species produces a unique, identifiable CGR pattern.

2.3) Proteins and the need for CGR

Proteins are large, complex molecules made up of amino acid chains, which fold into specific three-dimensional shapes to perform a wide range of functions in the body. These functions include catalysing biochemical reactions (enzymes), providing structural support (collagen), and regulating cellular processes (hormones). The sequence of amino acids in a protein, known as the primary structure, determines its shape and function.

CGR helps visualize protein sequences by transforming the linear sequence of amino acids into a fractal pattern, which can reveal underlying structural features and compositional trends. Protein sequences, composed of 20 amino acids rather than the four nucleotides in DNA, require a more complex mapping system. CGR allows us to quickly spot repetitive motifs or structural similarities, aiding in protein function prediction, structural analysis, and the detection of mutations or anomalies.

.

3) Methodology

The goal of this project is to provide a web-based platform where users can upload biological sequence files (in FASTA format) and generate a visual representation of the sequences using Chaos Game Representation (CGR). The application supports both DNA and protein sequences, offering users an interactive way to visualize the structure of these biological sequences.

The core features of the web application are:

- Uploading DNA or protein sequence files in FASTA format.
- Processing the sequence and generating CGRs for DNA and protein sequences.
- Displaying the generated CGR plot on the result page.

- **Importing the necessary libraries**

- Flask - a lightweight Python web framework used to build the web application. It handles routing, rendering HTML templates, and managing file uploads, making it ideal for creating an interactive web interface for generating and displaying CGR plots.

- BioPython - Used for reading and parsing biological sequence files in FASTA format. It simplifies the process of extracting DNA or protein sequences and ensures the correct format, enabling efficient sequence processing for the CGR generation.

- Matplotlib - A plotting library used to create the Chaos Game Representation (CGR) visualizations. It provides tools to generate high-quality plots and save them as image files, which are then displayed on the web interface.

- Werkzeug - Utility library that supports Flask, particularly for handling file uploads. It ensures that file names are sanitized before being saved, which adds a layer of security by preventing potentially harmful files from being processed.

- Numpy - It handles mathematical operations like coordinate transformations and midpoint calculations efficiently, making it essential for generating the dodecagon layout and visualizing protein sequences.

File upload and Processing

The application allows users to upload sequence files in FASTA format through a web interface. Users can select the type of sequence on which they have to perform CGR – either DNA or Proteins and then upload the file. Upon submission, the following process takes place:

(i) File Handling – The *secure_filename* function from the *werkzeug.utils* ensures that the file is save with a safe and sanitized name within the directory

(ii) Reading the FASTA file – Once the file is uploaded, the *Bio.SeqIO module* is used to parse the FASTA file and extracted using the *SeqIO.parse( ).* The sequence is converted to uppercase to ensure consistency and then passed into the corresponding CGR function

Chaos Game Representation of DNA

CGR for DNA sequences uses a square grid to represent the four nucleotides: Adenine (A), Cytosine (C), Guanine (G), and Thymine (T). Each nucleotide is mapped to a specific coordinate

- We begin the algorithm from the center of the square, which is (0.5, 0.5)
- This initial point represents a neutral starting location in the middle, giving the sequence a balanced starting point.

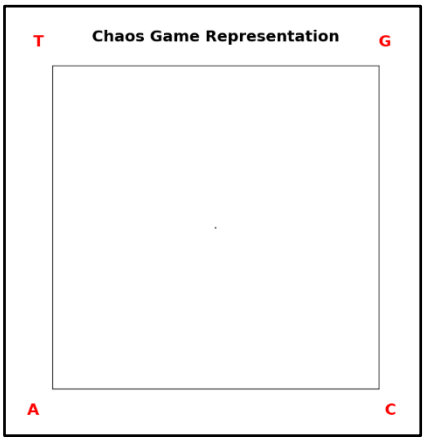


Fig 4 - CGR of DNA in square

```
- Initialize base mapping for DNA bases:
      - 'A' maps to (0, 0)
      - 'C' maps to (1, 0)
      - 'G' maps to (1, 1)
      - 'T' maps to (0, 1)

   - Set initial coordinates (x, y) to (0.5, 0.5) (start in the center)
   - x_coords =[ ]
   - y_coords =[ ]
      - Add (x, y) to x_coords and y_coords
```

Iteratively updating the position

For each base in the DNA sequence, we calculate a new coordinate based on the midpoint between the current position and the base-specific coordinate. This means each time we move halfway from the current position to the coordinate of the next base in the sequence and mark a new location.

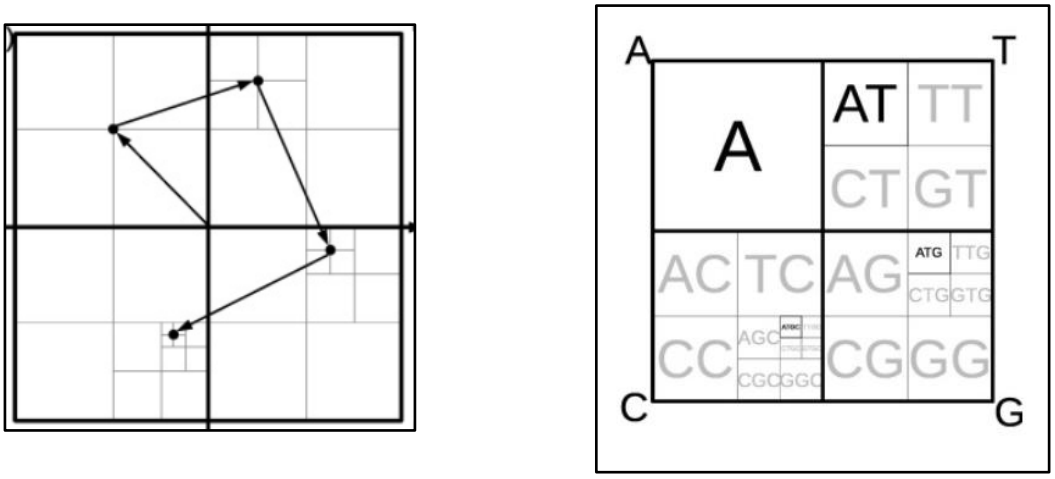


Fig 5 - Position update in DNA CGR

Updating the coordinate list

After calculating the new position (x, y), we add these coordinates to lists *x_coords* and *y_coords*. This allows for easy plotting once all points are computed. Over time, as we iterate through the sequence, these coordinates accumulate, forming a distinctive fractal pattern specific to the DNA sequence.

```
Iterate over each base in the DNA sequence:
    - If base is 'A', 'C', 'G', or 'T':
        - Get corresponding coordinates (base_x, base_y) from base mapping
        - Calculate new coordinates:
            - x = (x + base_x) / 2
            - y = (y + base_y) / 2
        - Append new coordinates (x, y) to x_coords and y_coords
```

Plotting the CGR point using Matplotlib

Once all points are calculated for each base in the sequence, they're plotted on a scatter plot to visualize the fractal structure. Points that are repeatedly drawn closer to particular corners (based on the base's frequency) create dense clusters, while less frequent bases create sparser areas. This unique distribution provides insights into the DNA sequence composition and base arrangement. In the final CGR plot, a single coordinate can encode the complete sequence input parsed till then.

Chaos Game Representation of Proteins

CGR of proteins is implemented a bit different from the previous one. In the amino acid-based CGR for protein sequences, one could theoretically assign each of the 20 amino acids to a unique vertex on a 20-sided polygon, with each point corresponding to one amino acid. However, this approach has two main drawbacks:

(i) Difficulty in Identifying similar pattern

- Proteins from the same family often contain conservative substitutions, where one amino acid is replaced by another similar one without impacting the protein's function.
- If each amino acid has its own vertex, then conservative substitutions will result in different patterns even for proteins with highly similar functions. This difference can obscure similarities between homologous proteins and make it challenging to group proteins based on functional or structural resemblance.

(ii) Overwhelming Complexity and Visual Clutter

- A 20-vertex structure would add complexity without significantly enhancing our understanding of sequence patterns.
- The high number of vertices increases the density of plotted points, making the resulting CGR visually cluttered and less interpretable.

Therefore, for CGR applied to protein sequences, the method must consider groups of amino acids rather than treating each amino acid as an individual point.

Adaptations made to address the issues :

- **Conservative Substitutions**: Amino acids that can replace each other in homologous proteins without changing the protein's function should be grouped together. For instance, alanine (A) and glycine (G), which are functionally similar, are represented as one vertex.

- **Reduced Vertex Count for Clarity**: Using a 12-sided polygon instead of 20 vertices (one for each amino acid) improves visual patterns and helps differentiate between homologous (functionally similar) and non-homologous proteins.

Grouping Amino Acids for the 12-Vertex CGR

To create a 12-vertex CGR for proteins, amino acids are grouped based on similarity in their chemical properties and their likelihood of conservative substitution in homologous proteins. This grouping reflects how similar amino acids tend to replace each other in protein sequences while preserving functionality.

- **Chemical and Structural Properties**: Amino acids with similar side chain properties (e.g., size, charge, hydrophobicity) are grouped together. For example, Isoleucine (I), Leucine (L), Valine (V), and Methionine (M) are all hydrophobic and structurally similar, so they are placed in the same group.
- **Conservative Substitution Patterns**: Amino acids that often replace each other in homologous sequences are grouped, as these substitutions generally do not affect protein function. For instance, Serine (S) and Threonine (T), both of which have hydroxyl groups, frequently substitute for one another in proteins without altering the protein's function.

Basis of grouping the amino acids

1. Alanine (A) and Glycine (G): Both have small side chains and flexible with similar structural roles, often found in turns and flexible regions.

2. Serine (S) and Threonine (T): Both have polar hydroxyl (-OH) groups in their side chains, allowing similar hydrogen bonding and functional roles in enzyme active sites.

3. Isoleucine (I), Leucine (L), Valine (V), Methionine (M): These are hydrophobic and have bulky side-chains, typically located in the protein's interior for structural stability and to avoid contact with water. They behave similar in protein folding too.

4. Aspartic Acid (D) and Glutamic Acid (E): Both are negatively charged at physiological pH due to their carboxyl (-COOH) , playing similar roles in ionic interactions and enzyme catalysis.

5. Lysine (K) and Arginine (R): Both have positively charged side due to amino (-NH2) groups, contributing to similar ionic interactions and salt bridge formation.

6. Asparagine (N) and Glutamine (Q): Both have polar amide side chains (-CONH2), involved in hydrogen bonding and important for protein stability.

7. Phenylalanine (F), Tyrosine (Y), and Tryptophan (W): These aromatic amino acids are hydrophobic and involved in stabilizing structures through π-π interactions and hydrophobic forces.

8. Histidine (H): Important in binding and catalysis due to its ability to carry a charge at certain pH levels.

9. Proline (P): Its rigid, cyclic structure causes bends in the protein chain, helping in protein folding.

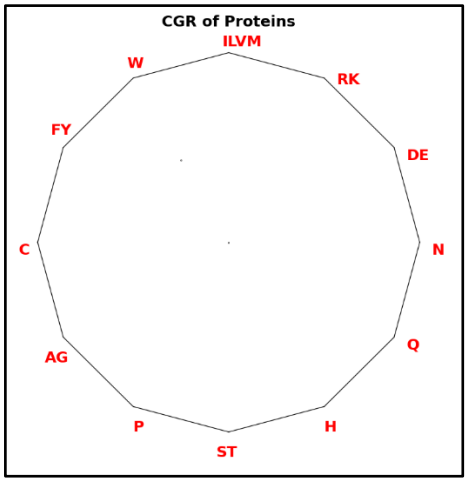
10. Cysteine (C): Cysteine has a thiol (-SH) group, crucial for forming disulfide bonds that stabilize protein structures.

11. Glutamic Acid (E) and Aspartic Acid (D): These acidic amino acids have similar chemical properties, playing key roles in maintaining protein charge and interactions.

12. Arginine (R) and Lysine (K): Both are basic amino acids with similar roles in protein interactions and catalysis due to their positively charged side chains.

Mapping the groups into vertices

Group functionally similar amino acids (conservative substitutions) into 12 clusters, with each group represented by a single vertex on a 12-sided polygon. (dodecahedron). Initialize the plot at the polygon's centre i.e (0.5, 0.5), establishing a neutral starting point.



Calculating and storing for position update

- For each amino acid in the protein sequence, we calculate a new coordinate based on the midpoint between the current position and the specific coordinate. This means each time we move halfway from the current position to the coordinate of the next amino acid in the sequence and mark a new location.
- Plot the stored coordinates using a scatter plot

4) Experiments

- **Dataset Selection**

- **Source and Format:** The dataset for this experiment was obtained from the National Center for Biotechnology Information (NCBI) database, one of the most reputable and widely used resources for biological sequence data. NCBI provides extensive collections of nucleotide and protein sequences, offering public access to FASTA-formatted files suitable for computational analyses.
- This dataset comprises DNA sequences specifically chosen for the analysis of Chaos Game Representation (CGR). The FASTA format was selected as it is straightforward, listing each sequence identifier in a header line followed by the nucleotide sequences, which makes it compatible with various bioinformatics tools.
- **Purpose of Selection:** These DNA sequences were selected to study and visualize structural patterns within CGR plots. By analysing these sequences, we aim to identify potential distinguishing features within the CGR that may be characteristic of specific genomic regions, aiding in understanding genomic diversity and complexity across different species.

- **Dataset Size and Composition**

- The length of DNA sequences in this dataset varies significantly, with sequence lengths ranging from as short as 1,500 nucleotides to as long as 40,000 nucleotides.
- This variability in length enables the analysis of both short and long genomic regions, allowing for insights into how sequence length may influence CGR patterns. Shorter sequences can offer a more focused view of localized motifs, while longer sequences provide a broader view of CGR patterns over more extensive genomic regions.
- The length of protein sequences ranges for about 500 to 800 amino acids

Diversity of Genetic Material:

- The dataset includes sequences from various organisms, which could cover a range of genetic functions and structural characteristics.
- This diversity is crucial for examining the efficacy of CGR in differentiating between distinct types of genetic material, such as coding regions (genes) versus non-coding regions (introns and regulatory sequences).
- The inclusion of both conserved and variable regions across species enhances the potential to detect unique CGR motifs associated with functional or evolutionary constraints.

Brief Explanation of the CGR algorithm

- Chaos Game Representation (CGR) is a mathematical technique for transforming a linear sequence into a fractal-like image, providing a visual representation that reveals patterns within the sequence.
- Originally developed for analyzing DNA sequences, CGR has been adapted to analyze both nucleotide and protein sequences.
- The CGR algorithm works by iteratively plotting points within a geometric shape, such as a square for DNA sequences (representing the four nucleotide bases) or a dodecagon for proteins (representing groups of amino acids).
- The result is a spatial distribution of points that can reveal recurring patterns, motifs, and structure in the sequence

Application of the CGR algorithm

(i) Pattern Recognition in Genomic Sequences:

- CGR is useful for identifying motifs or repeats in DNA.
- For instance, repetitive sequences form recognizable patterns in the CGR plot, aiding in the detection of genetic elements associated with specific biological functions or diseases.
- CGR can also differentiate coding regions (exons) from non-coding regions (introns or regulatory elements) based on the patterns that these regions produce in the CGR plot, as coding regions tend to have more specific structural motifs due to evolutionary constraints**.** For instance, areas with high codon repetition (like ATG or TAA) will generate dense clusters in certain vertices of the CGR shape, revealing these conserved sequences.

(ii) Classification of Protein Families:

- By applying CGR to protein sequences, one can group similar amino acids based on their properties (e.g., hydrophobicity or charge) and observe patterns in the resulting CGR plot.
- This approach can reveal the structural motifs common to proteins within the same family, providing insight into functional similarity.
- This visualization is particularly useful in identifying conserved regions across homologous proteins, even if they include conservative amino acid substitutions. By categorizing amino acids into groups, CGR can show similar patterns for sequences that are functionally similar despite minor sequence differences**.**

Advantages of CGR for Sequence Analysis

- Scalability: CGR is effective for analyzing both small segments and large-scale sequences, enabling it to scale well across datasets with varying lengths.
- Pattern Detection and Visualization: CGR's fractal approach provides a unique way to visualize recurring patterns and motifs within sequences, highlighting features that may not be immediately obvious in linear representations.
- Enhanced Understanding of Sequence Structure: Through the geometric arrangement of sequence motifs, CGR provides insights into the structural organization of DNA and proteins, which can be critical for understanding gene regulation, protein folding, and evolutionary biology.

5) Results and Discussions

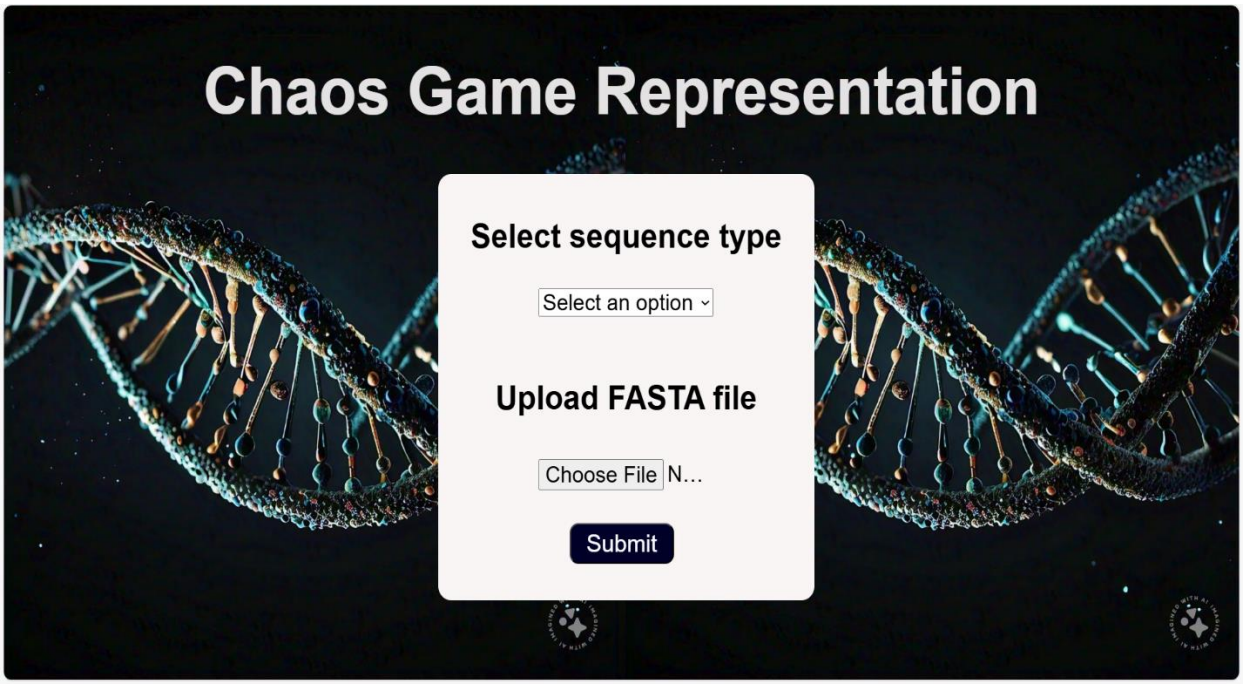


Fig 6 - GUI of the web application

- The user can choose the type of sequence on which the CGR is to be performed – DNA/Protein
- The user can then upload the sequence file in FASTA format and submit

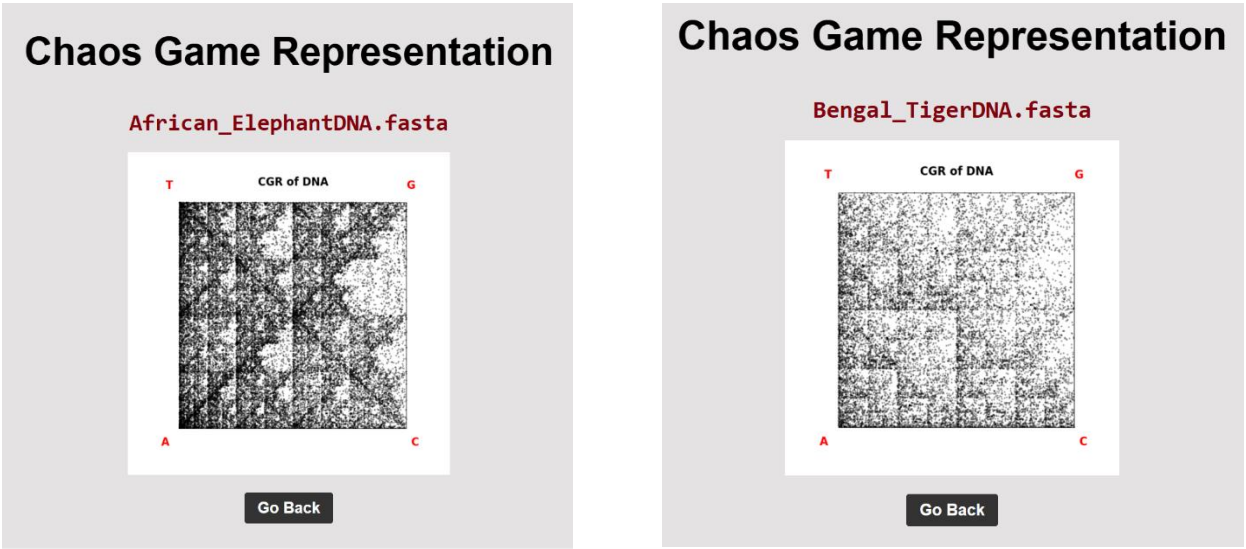


Fig 7 - DNA of African Elephant (left) and Bengal Tiger (right)

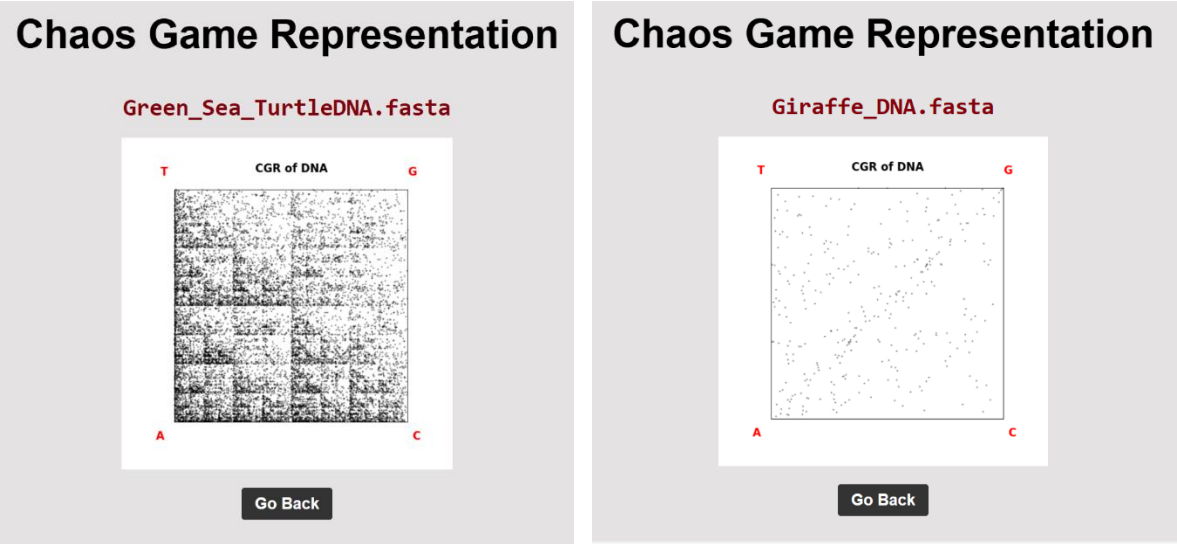


Fig 8 - DNA of Green Sea Turtle (left) and Giraffe (right)

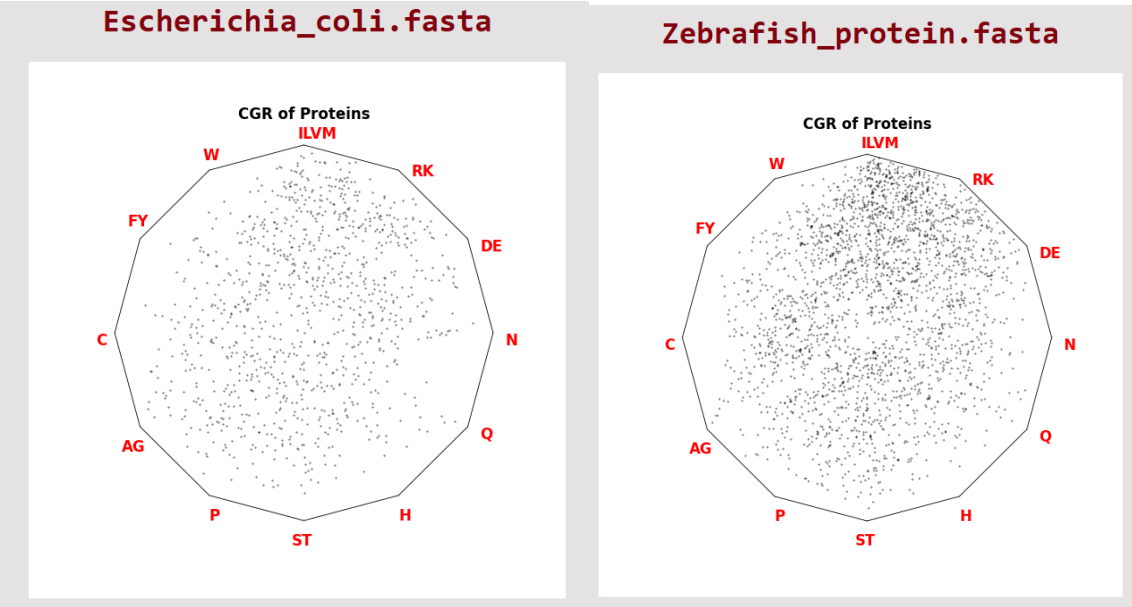


Fig 9 - Proteins of E-coli (left) and Zebrafish (right)

INFERENCES

(i) Identification of Coding and Non-Coding Regions

- Coding (exons) and non-coding (introns, regulatory elements) regions exhibit distinct patterns.
- Coding regions often display more organized, repetitive motifs due to the conservation of codon triplets required for protein synthesis, while non-coding regions show less structured, more random distributions.
- In a CGR plot, coding regions form dense, recognizable clusters because of their repetitive and structured nucleotide sequences. In contrast, non-coding regions tend to generate more dispersed and less dense patterns

(ii) Detection of Palindromic and Repetitive Sequences

- DNA sequences with palindromic (mirror-like) or repetitive patterns can be visualized clearly in CGR plots. These elements play crucial roles in genetic regulation, replication, and stability.
- Palindromic sequences generate symmetrical patterns in the CGR plot, as the repetition and mirroring in nucleotide order cause the plotted points to mirror along specific axes.
- Repetitive sequences also appear as recurring clusters, providing visual cues about these highly repetitive regions.

(iii) Identifying Species-Specific Signatures

- Different organisms and genomes often have distinct CGR patterns due to their unique nucleotide composition and organizational structure, which can serve as a "genomic fingerprint."
- By comparing CGR plots across species or genome types, we can identify species-specific traits, such as GC content bias, codon usage, and other sequence characteristics.

(iv) Identification of Mutations

- Mutations, especially high-frequency SNPs, alter the sequence in specific ways that can cause noticeable deviations in the CGR plot from what is expected for a "normal" or reference sequence.

- When comparing CGR plots of a sequence with a known reference, regions where patterns deviate may indicate potential mutations or polymorphisms.

6) Conclusion

This project utilized the Chaos Game Representation (CGR) technique to visualize both DNA and protein sequences, enabling us to examine the underlying structure and patterns of these biomolecules. CGR provides a graphical representation that transforms sequences into fractal-like patterns, giving insights into their composition, organization, and functional regions.

- For DNA sequences, CGR helped us to identify patterns in nucleotide distribution, repetitive regions, and structural motifs.
- Through the CGR plots, can observe nucleotide composition, detection of motifs and repeats and genomic fingerprinting.
- For protein sequences, CGR offers a different perspective, as the plot reflects amino acid groupings based on properties like polarity, charge, and hydrophobicity.

7) Future Work

The application of CGR to both DNA and protein sequences opens numerous possibilities for further research and improvement.

- **Enhanced Comparative Genomic and Proteomic Studies**: By applying CGR to large datasets of DNA and protein sequences, we can perform comparative analyses across species. This can help reveal conserved or divergent sequence characteristics, shedding light on evolutionary relationships, functional similarities, and adaptations.
- **Integration of Machine Learning for Pattern Recognition**: Machine learning models, such as convolutional neural networks (CNNs), could be trained on CGR plots to detect specific patterns, functional motifs, or sequence signatures in both DNA and protein sequences. This approach would enhance the scalability of CGR analysis, enabling automatic detection of genes, protein domains, or conserved motifs.
- **Comparing Genetic Similarity Between Organisms**: By comparing CGR plots of DNA or protein sequences from different organisms, we can assess the genetic similarity or evolutionary distance between them. This comparison could help determine how closely related two organisms are at the genomic or proteomic level, aiding in the identification of common ancestors, divergent evolutionary paths, or species classification.

References

[1] Soumalee Basu, Archana Pan, Chitra Dutta, Jyotirmoy Das (1997), *Chaos game representation of proteins, Journal of Molecular Graphics and Modelling*
https://www.sciencedirect.com/science/article/pii/S109332639700106X

[2] Cristina Stan, Constantin P. Cristescu, Eugen I. Scarlat (2010) , *Similarity analysis for DNA sequences based on chaos game representation. Case study: The albumin, Journal of Theoretical Biology*
https://www.sciencedirect.com/science/article/pii/S0022519310005011

[3] Hannah Franziska Löchel, Dominik Heider (2021), *Chaos game representation and its applications in bioinformatics, Computational and Structural Biotechnology Journal*
https://www.sciencedirect.com/science/article/pii/S2001037021004736

[4] Jian-Yi Yang, Zhen-Ling Peng, Zu-Guo Yu, Rui-Jie Zhang, Vo Anh, Desheng Wang (2009), *Prediction of protein structural classes by recurrence quantification analysis based on chaos game representation, Journal of Theoretical Biology*
https://www.sciencedirect.com/science/article/pii/S0022519308006723

[5] Jonas S. Almeida, João A. Carriço, António Maretzek, Peter A. Noble, Madilyn Fletcher (2001), *Analysis of genomic sequences by Chaos Game Representation , Bioinformatics*
https://academic.oup.com/bioinformatics/article/17/5/429/277428