Bioinformatics, 2024, pp. 1–1

doi: DOI HERE
Advance Access Publication Date: Day Month Year
Supplementary

SUPPLEMENTARY

fcgr: A Python library for accurate Frequency Chaos Game Representation of DNA sequences

Abhishek Halder[1,*] Piyush[2] Bernadette Mathew[3] and Debarka Sengupta[4]

[1]Department of Computer Science and Engineering, Indraprastha Institute of Technology, 110020, Delhi, India, [2]Department of Computer Science and Engineering, Dronacharya College of Engineering, 123506, Haryana, India, [3]Department of Computational Biology, Indraprastha Institute of Technology, 110020, Delhi, India and [4]Department of Computer Science and Engineering, Indraprastha Institute of Technology, 110020, Delhi, India
[*]Corresponding author. debarka@iiitd.ac.in
FOR PUBLISHER ONLY Received on Date Month Year; revised on Date Month Year; accepted on Date Month Year

Abstract

Motivation: Frequency Chaos Game Representation (FCGR) is an effective method for encoding DNA sequences by counting the frequency of each k-mer in a predefined manner. It is used in alignment-free genomic comparison, protein sequence feature extraction, motif finding, antimicrobial resistance classification, and phylogenetic analyses. However, current FCGR implementations in R often produce inaccurate results and scale poorly with larger DNA sequences.

Results: We present *fcgr*, a Python package that constructs the FCGR representation of DNA sequences with improved speed and accuracy. It offers functionalities for reading FASTA files, querying k-mer positions, generating k-mer frequency matrix and dictionary with optional pseudo-count adjustments.

Availability: The *fcgr* Python package, along with documentation, is available on GitHub for non-commercial use.

Contact: For inquiries, please contact debarka@iiitd.ac.in.

Supplementary: Supplementary data can be found at GitHub.

Derivation of the FCGR Matrix from DNA sequence via CGR

Algorithm 1 Derivation of the FCGR Matrix from DNA sequence via CGR

1: **Input:** DNA sequence, k-mer length (k)
2: **Output:** FCGR matrix
3: **Matrix and Grid Initialization:**

- Configure the FCGR matrix of size $2^k \times 2^k$.
- Initialize each cell of the FCGR matrix to zero.
- Assign specific coordinates to nucleotides:
  - Guanine (G) at $(-1, -1)$
  - Thymine (T) at $(-1, 1)$
  - Adenine (A) at $(1, -1)$
  - Cytosine (C) at $(1, 1)$

4: **Walker Simulation:**

- Initialize the walker at the origin $(0, 0)$.
- Sequentially process each nucleotide in the DNA sequence from left to right:
  - Identify the corresponding vertex for the nucleotide.
  - Calculate the midpoint between the walker's current position and the corresponding vertex.
  - Move the walker to this midpoint and update its position.
  - If the number of processed nucleotides is at least k, locate the grid cell coordinate based on the walker's current position and increment the count in the corresponding cell of the FCGR matrix by one.

5: **Result Compilation:**

- Output the completed FCGR matrix, which quantifies the frequency of each k-mer encountered by the walker.