CS466 – Introduction to Bioinformatics
Department of Computer Science
Spring 2021
Grainger College of Engineering, University of Illinois at Urbana–Champaign

---

Mini Project Report

Motif Finding Methods

Dr. Saurabh Sinha

Team Members

Liu, Pan	*panl3@illinois.edu*
Zhu, Ruike	*ruikez2@illinois.edu*

1. INTRODUCTION

A 'motif' is a pattern in a sequence. For example, in DNA sequences (which are sequences over the alphabet A, C, G, T), an example of a motif is the pattern 'TCACGTG'. A slightly more complex motif is the pattern TC[A/C]CGTG, which represents 'either TCACGTG or TCCCGTG'. An occurrence of a motif in a given DNA sequence is called a 'site'.

A more popular form of a motif is that of the 'position weight matrix' (PWM). This is a probabilistic pattern. An example of a PWM (with motif length = 8, sequence alphabet = A, G, C, T - length = 4) is shown below:

$$\begin{bmatrix} 0.6 & 0.4 & 0 & 0 \\ 0 & 0.8 & 0.2 & 0 \\ 0.2 & 0 & 0.8 & 0 \\ 0 & 0 & 0 & 1 \\ 0.6 & 0.2 & 0 & 0.2 \\ 0.2 & 0.8 & 0 & 0 \\ 0.2 & 0 & 0.6 & 0.2 \\ 0 & 0 & 0.2 & 0.8 \end{bmatrix}$$

The 'information content' of a PWM 'W' of length L is defined by:

$$IC = \sum_{k} \sum_{\beta \in \{A,C,G,T\}} W_{\beta k} log(\frac{W_{\beta k}}{q_{\beta}})$$

where $W_{\beta k}$ = probability of base β at position k and q_{β} = probability of base β by chance ('background')

The information content represents how 'sharp' the pattern is. For example, if every position is uniformly distributed among the 4 characters, the information content is 0. (Check this for yourself.) If a position prescribes an 'A' with probability 1 and all other characters are disallowed (probability 0), that position contributes log(4) to the information content, the maximum possible contribution of a single position of the motif.

Despite considerable efforts to date, DNA motif finding remains a complex challenge for biologists and computer scientists. Researchers have taken many different approaches in developing motif discovery tools and the progress made in this area of research is very encouraging. Most of the earlier literature categorized motif finding algorithms into two major groups based on the combinatorial approach used in their design: (1) word-based (string-based) methods that mostly rely on exhaustive enumeration, i.e., counting and comparing oligonucleotide

frequencies and (2) probabilistic sequence models where the model parameters are estimated using maximum-likelihood principle or Bayesian inference [1].

2. PROBLEM STATEMENT

The project involves developing a "motif finding" program and testing it. The goal of motif finding is the detection of unknown signals in a set of DNA sequences. In our project, given generated sequences, the program is going to find expectation of motifs and sites. The three major components of the implementation are:

- Building a benchmark
- Implementing the "motif finder"
- Evaluating the motif finder on the benchmark and making intelligent inferences.

We will use the Greedy Motif Search algorithm introduced in the lecture since it is relatively easy to implement and time-saving.

3. DESCRIPTION OF GREEDY ALGORITHM

The proposed greedy motif search algorithm, **GreedyMotifSearch**, does not use any of the aforementioned tree traversals because it is not an exhaustive search method. However, the greedy method does do an exhaustive search on the first two strands of DNA to determine the best motif in these two strands. This motif is called the seed. The method then sequentially searches the remaining DNA strands for the motif in each strand that best matches the seed and the motifs that have already been found. The score used for evaluation in the algorithm is information content. In short, the method is trying to find PWM with the greatest information content. The algorithm is it is not guaranteed to find the optimal solution.

The whole process can be summarized as follows:

- Find two *l*-mers in sequences 1 and 2, form $2 \times l$ alignment matrix and compute Score(s); pick the s with the highest information content.
- Iteratively add one *l*-mers from each of the other (t-2) sequences
- At each of the following t-2 iterations, finds a "best" *l*-mers in sequence i. That is, try each *l*-mers in sequence i, add it to profile matrix, compute score, and pick the *l*-mers that leads to the highest score.
- Sacrifices optimal solution for speed: in fact the bulk of the time is actually spent locating the first 2 *l*-mers

3.1 High Level Pseudocode

Greedy Motif Search Algorithm to predict motif (position weight matrix) and predict sites in DNA sequences

Input: motiflength.txt, sequences.fa

Output: predictedmotif.txt, predictedsites.txt

Algorithm GreedyMotifSearch

```
bestMotif ← (1,1,…,1)
s ← (1,1,…,1)
for s_1 ← 1 to n – l + 1
        for s_2 ← 1 to n – l + 1
        if Score(s,2,DNA) > Score(bestMotif, 2, DNA)
                BestMotif_1 ← s_1
                BestMotif_2 ← s_2
s_1 ← BestMotif_1
s_2 ← BestMotif_2 1
for i ← 3 to t
        for s_1 ← 1to n – l + 1
                if Score (s, i, DNA) > Score (bestMotif, i, DNA)
                        BestMotif_i ← s_i
        s_i ← BestMotif_i
return bestMotif
```

3.2 External Libraries Used in Program

- **SciPy**: scipy.special.rel_entr function is used in evaluate.py to calculate the relative entropy between motif.txt and predictedmotif.txt
- **NumPy**: NumPy is used to manipulate arrays and generate random numbers for sequence generation.

3.3 Complexity

The complexity of the greedy algorithm is $O(ln^2 + nlt)$ where l is the length of the motif, n is the length of the DNA samples, and t is the number of DNA samples. So, this method has a squared term for the exhaustive search of the first two DNA strands, and then the rest of the program is a linear search.

REFERENCES

[1] Das, M.K., Dai, HK. A survey of DNA motif finding algorithms. *BMC Bioinformatics* **8,** S21 (2007). https://doi.org/10.1186/1471-2105-8-S7-S21