Marmara University
Faculty of Engineering

CSE 4065
Introduction to Computational Genomics

Assignment # 1

Instructor: Betül BOZ   **Due:** 07.04.2025

	Department	Student Id Number	Name & Surname
1	CSE	150120066	Zeynep YILMAZ
2	CSE	150121077	Efe ÖZGEN
3	CSE	150120070	Semih BAĞ

1. Introduction

The main goal of this assignment is to implement two widely used motif-finding algorithms, **Randomized Motif Search** and **Gibbs Sampler**, and analyze their performance. Both algorithms were implemented in Java and tested with different motif lengths. Their results were compared in terms of motif similarity, accuracy, and execution time.

2. Implementation

2.1 Creating Input

We generated 10 random DNA strings, each of length 500. A random k-mer of length *k* was created and mutated at 4 random positions to produce 10 different versions. Each mutated k-mer was then inserted into a random position within each DNA string. The final sequences were saved into input.txt.

2.2 Randomized Motif Search

Randomized Motif Search provides a fast way to approximate good motifs, but it does not guarantee finding the optimal solution.

RandomizedMotifSearch(DNA, k, t):

  1. Randomly select k-mers Motifs = (Motif$_1$, ..., Motif_t) from each DNA string

  2. BestMotifs ← Motifs

  3. Repeat:

    a. Construct Profile matrix from Motifs with pseudocounts

    b. For each string in DNA:

      Select the most probable k-mer using the Profile

    c. Motifs ← newly selected k-mers

    d. If *Score*(Motifs) < *Score*(BestMotifs):

      BestMotifs ← Motifs

    Else:

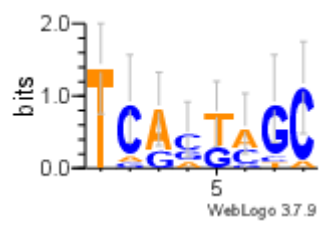   Return BestMotifs

2.3 Gibbs Sampler

Gibbs Sampler generally performs better than Randomized Motif Search in finding lower-scoring motifs. However, it does not guarantee the optimal solution and usually requires more time, creating a trade-off between accuracy and runtime.

GibbsSampler(DNA, k, t, N):

   1. Randomly select k-mers Motifs = (Motif$_1$, ..., Motif_t) from each DNA string

   2. BestMotifs ← Motifs

   3. For i = 1 to N:

      a. Randomly select one sequence index j ∈ [1, t]

      b. Remove Motif_j from Motifs

      c. Construct Profile from remaining Motifs with pseudocounts

      d. Calculate probability Pr(k-mer | Profile) for each k-mer in DNA[j]

      e. Choose a new k-mer from DNA[j] with probability proportional to Pr

      f. Insert the selected k-mer back into Motifs at position j

      g. If *Score*(Motifs) < *Score*(BestMotifs):
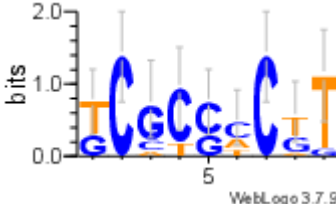
         BestMotifs ← Motifs

   4. Return BestMotifs

---

3. Results

k = 8 , `RandomizedMotifSearch`
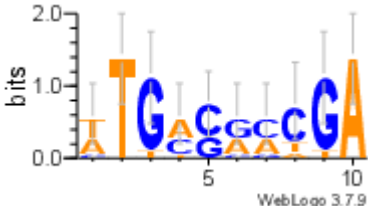
WebLogo	
Motifs	TCGCGAGC TCAATAGA TGACTCGC TCAGTAGC TCACGGGC TCAAGAGC TCGGGCGC TCAGTCCC TAGCTCGC

	TCACTATC
Time	15.808 ms
Consensus String	TCACTAGC
Best Score	22.0
Avg Score	22.30

k = 8, `GibbsSampler`

WebLogo	
Motifs	GTAGATAT ATAGATAT GTCGTTAT GTAGATTG CTATGTAT GAAGATTT ATCTATCT GTCCATTT GTAGTTAT GTAGGTTT
Time	6.9711 ms
Consensus String	GTAGATAT
Best Score	20.0
Avg Score	25.42

k = 9, `RandomizedMotifSearch`

WebLogo	
Motifs	CCCGTGAAG CGGGGGTAC CGAGAGAAG CGACTGAAC CTAGTAAAC CTGCTGAAG CTGGTGAAC

	CAAGTGAAG CGCGTGAAG CGGGTAAAC
Time	6.9649 ms
Consensus String	CGAGTGAAC
Best Score	23.0
Avg Score	23.84

k = 9 , GibbsSampler

WebLogo	
Motifs	TCGTGCCGT TCGCCTCGT TCGCGACGT GCACCTCTT TCGCCCCAT GCCCCCCTT GCGCGCCTG TCGCCACTT GCCCGCCTT TCGTCACGT
Time	1.5685 ms
Consensus String	TCGCCCCTT
Best Score	24.0
Avg Score	27.80

k = 10 , RandomizedMotifSearch

WebLogo	
Motifs	ATGACGCCGA ATGCGAACGA TTTACGCCGA ATGCGACTGA TTGCGGGCGA

	TTGAGACCGA TTGCCCACGA TTGTCAATGA ATGACGCATA CTGACGACGA
Time	10.1101 ms
Consensus String	TTGACGCCGA
Best Score	29.0
Avg Score	29.53846153846154

k = 10 , `GibbsSampler`

WebLogo	
Motifs	TGACGCCGAG TGATGCCCGA TGACTCGCGA AGACGCGCGA TAATTCAGAA TGAAGCATAA TGAATCACGG TGATTCGGGG TGACGCATAG TGATTCACGA
Time	2.6849 ms
Consensus String	TGACGCACGA
Best Score	30.0
Avg Score	37.21

4. Discussion

In this study, we analyzed and compared the performance of two widely used motif-finding algorithms: Gibbs Sampler and Randomized Motif Search (RMS). Our primary goal was to determine which algorithm provides more accurate motif approximations for varying lengths of k (8, 9, and 10). Each k-mer value had its unique generated motif, and each algorithm's performance was assessed based on motif accuracy, execution time, and consistency across multiple runs.

For k=8, Gibbs Sampler demonstrated superior performance compared to Randomized Motif Search. Gibbs sampler consistently found better motifs, achieving lower motif scores and consensus strings that were slightly closer to the generated motif. An important observation was that Gibbs Sampler ran faster than RMS for this k-mer value, suggesting higher efficiency for smaller motif sizes. Conversely, RMS required more execution time and generally produced motifs further from the generated sequence.

When we increased the motif length to k=9, the results changed. RMS slightly outperformed Gibbs sampler in terms of finding lower-score motifs. However, despite RMS having better scores, neither of the algorithms could approximate motifs close to the original generated k-mer. Interestingly, although Gibbs Sampler yielded worse scores, it executed significantly faster. This outcome highlights a notable tradeoff between execution time and motif approximation accuracy.

For k=10, RMS again achieved slightly better motif scores compared to Gibbs Sampler, yet both algorithms continued to struggle significantly in approximating the original motif. Gibbs Sampler again had a clear advantage in execution time. Despite RMS showing slightly better scores, both algorithms' consensus sequences remained notably distant from the generated motif, indicating the inherent difficulty in accurately finding larger motifs.

Additionally, we observed increased motif scores as k increased, meaning motif approximation quality declined with longer motifs. One reason for this deterioration is likely the increased complexity and the number of symbols evaluated, introducing more variations and making accurate approximations challenging. We also noticed that the consensus strings from different runs remained similar, even when their scores differed. This suggests that both algorithms' accuracy might heavily depend on their initial randomly selected motifs.

Overall, Gibbs Sampler showed better performance in terms of runtime and, at smaller motif sizes, better approximation accuracy. However, both algorithms showed considerable limitations for larger motif sizes, emphasizing the need for better starting point selection strategies or more robust algorithms for larger k-mers.

5. Conclusion

Based on our analysis, Gibbs Sampler generally outperformed Randomized Motif Search for smaller motif lengths (k=8), offering better motif approximations and faster execution. For larger motif lengths (k=9, 10), RMS yielded slightly better scores but both algorithms failed to accurately identify the generated motifs. Execution time differences were minimal (below 20

ms) for both algorithms, indicating speed is negligible in practical scenarios. Therefore, for smaller motifs, Gibbs Sampler is preferable, while for larger motifs neither algorithm demonstrated clear effectiveness. Future research should explore alternative motif discovery algorithms or enhanced strategies, particularly for larger k-mer sizes, to improve motif approximation accuracy.