# Contents

1 – Input File Creation and Insertion of K-Mer Motifs

2 – Randomized Motif Search

3 – Gibbs Sampler

4 – Program outputs

5 – Conclusion

In this project, we implemented two motifs finding algorithms Randomized Motif Search and Gibbs Sampler. And with the help of these algorithms, we tried to find consensus string from given DNA sequences.

1 – Input File Creation and Insertion of K-Mer Motifs

First, we created input file which contains the 10 DNA sequence each with the length of 500 characters long generated by using https://www.bioinformatics.org/sms2/random\_dna.html website. We inserted 10-mer motif AAAAAAAAAA with 4 mutations in random positions into

10 DNA sequences.

10-mer motifs with 4 mutations as follows:

ACCAATAGAA

AACTGAACAA

GACAATACAA

AAGACTAATA

ATAGACAGAA

GAAACATAAT

ACAAGAGAGA

AATTAGAAAC

AATGGATAAA

AGAATGCAAA

2 – Randomized Motif Search

Code snippet below is from the Main method where we start to execute our algorithms. We run the randomized motif search algorithm 3500 times. Because randomized motif search ends when it does not enhance the score. It means that it might stuck at local optimum. Thus,

We run the algorithm again and again to obtain the best consensus k-mer from the low score motifs.

And then, we put the returning best motifs into the HashMap with its score. After that we sort the HashMap ascending order by score value. And we select the lowest score value motifs which are the best motifs among all experiments.

Text

Description automatically generated

The method inside the below shows the implementation of the randomized motif search. First, we select the random k-mer motifs from sequences. And then, we create profile matrix from these motifs. We select new k-mer motifs with high score depending on calculation by using profile matrix. Finally, algorithm ends when it cannot improve the motif score and return the best motifs. Also, we use pseudocount while creating profile matrix.

Text

Description automatically generated

3 - Gibbs Sampler

We run Gibbs Sampler algorithm for 3500 times. For each time, we stop the algorithm, if score of the motifs is not changed last 50 iterations. We store these 3500 outputs in a hash map to find the motif with lowest score.

We used same methods from randomized motif search such for calculating scores, creating profile matrix and selecting k-mer from DNA. Firstly, we select randomly 10 k-mer from DNA. Then, we select one of them to remove from motifs. With these 9 sequences we create profile matrix. Most probably string is calculated for each line help of that profile matrix and is inserted to motifs. Best motifs are updated if these motifs’ score is lower than currently best motifs. Algorithm repeats this loop until no update occurs in last 50 iterations.



4 – Program outputs

Randomized motif search algorithm outputs as follows: For each k value, we run the algorithm 3500 times and take the lowest score(motifs) value motifs.

For k = 9

Chart

Description automatically generated

A computer screen capture

Description automatically generated with medium confidence

For k = 10

ATCTTGTTGA, ATCTACTAGA, CTCTCGCAAA, CTCTAGGAGA, CTCCTGTAGA, ATCTAAGAGA, CTCTAGTGCC, CTCTAATAGA, CTCTAGTACA, CTCTAGTAGC] = 20.0

Diagram

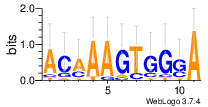
Description automatically generated

A computer screen capture

Description automatically generated with low confidence

For k = 11

ACAAAGTCTGA, ACAAAGTCCGA, AAAAGGTGGGA, AGAAAGCGGGA, ACCAAGCGGCA, CCAAAGTTGCA, TCCAACTGGCA, ACTAAGTGGGA, ACCTAGTGGGA, AGAAAATGGTA]=24.0,



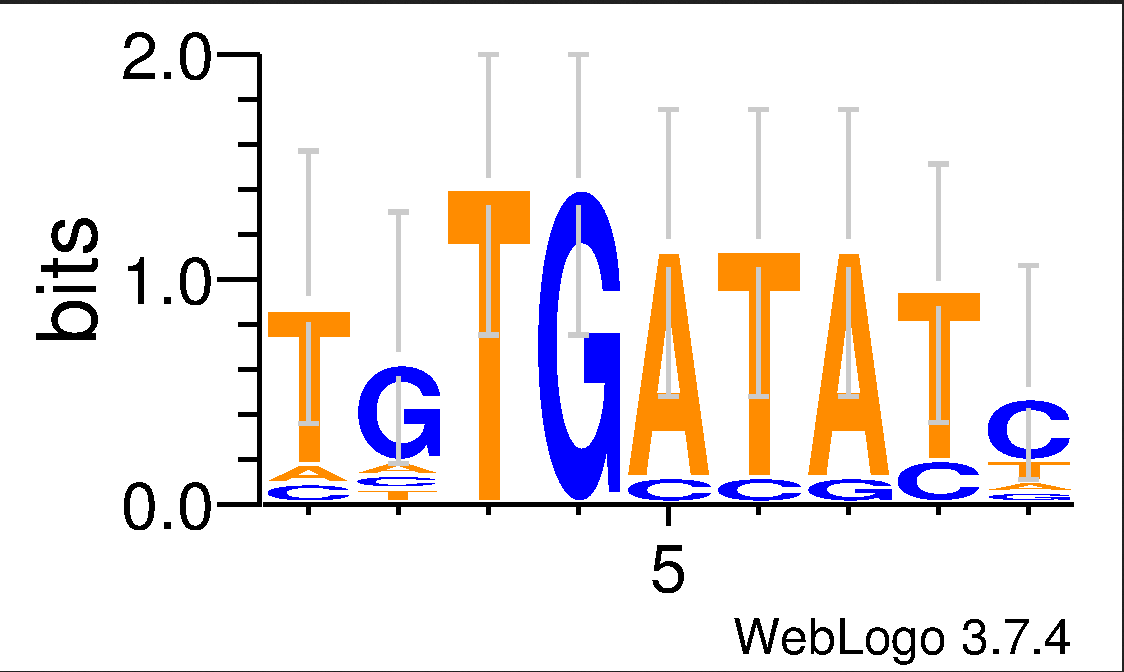
A computer screen capture

Description automatically generated with low confidence

Outputs of Gibbs Sampler

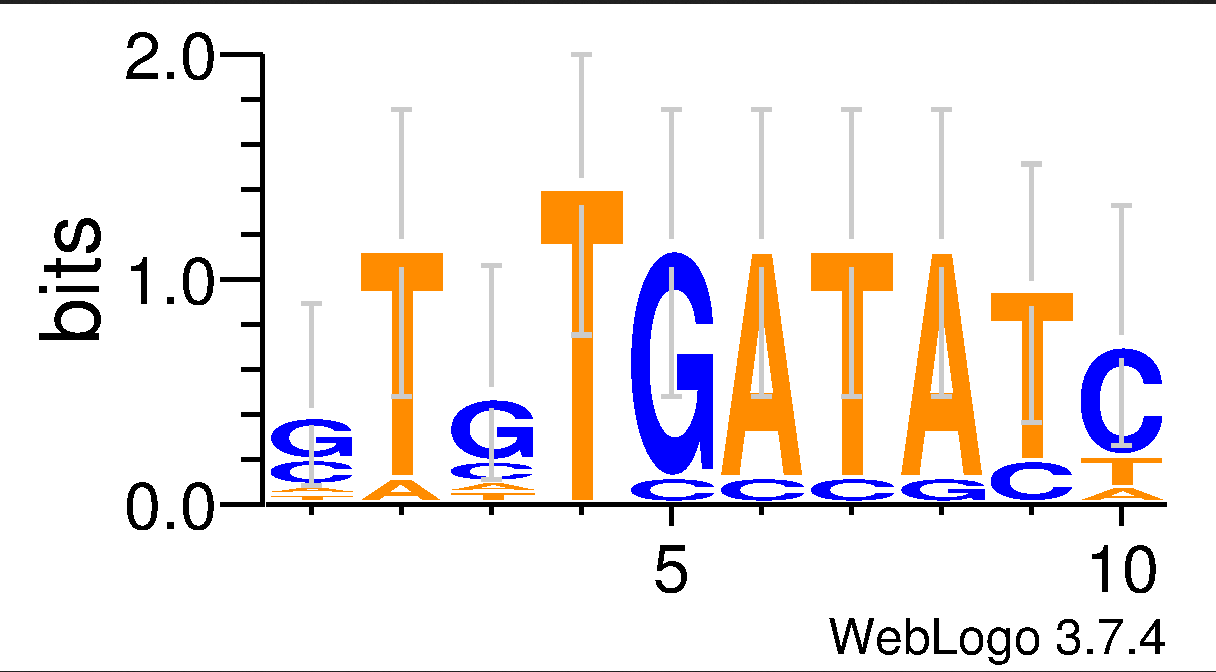
for k = 9

[TGTGACATC, TGTGATATT, TGTGCTACC, AGTGATATC, TGTGATATA, TGTGATACC, CGTGATATG, TATGATATC, TTTGATATT, TCTGATGTC]



for k = 10

[CTGTGACATC, ATGTGATATT, GTGTGCTACC, GAGTGATATC, CTGTGATATA, GTGTGATACC, GTCTCATATC, GTATGATATC, TTTTGATATT, CTCTGATGTC]

 for k = 11

[GGTGTTGATGT, GGAGGCGGCGT, GGACGCGAGGT, GCATTCGAGGT, GGACACGAGGT, CGGGTCGAGGT, GGAGTTAACGT, GGTCTGGAGGT, GGGATCGAGGT, GGAGTCGGGGG]

