MARMARA UNIVERSITY – FACULTY OF ENGINEERING

CSE 4065 ASSIGNMENT #1 REPORT

Name, Surname: Hasan Şenyurt – Melisa Durmuş – Mustafa Kerem Ekinci

Student ID: 150120531 – 150119727 – 150119695

Department: Computer Engineering

Input File

First, we created an input file containing the “A, T, G, C” nucleotides. When creating this input file, it is random to have 10 lines and 500 nucleotides per line. we defined our motif string as “ATCGGCTATT”. 4 mutations were inserted to random positions of this 10-mer. We placed the selected string in random positions on each line. We kept the indexes of the mutated strings in order to compare the results of the experiments (could it find the mutated string) and to make sure that the input file is correct.

Line by line mutated indexes;

1) String; “ATCCGATTTA”, index: 93
2) String; “ACCGGCTTGA”, index: 62
3) String; “ATTGGTAATA”, index: 68
4) String; “CCCGGTTATC”, index: 140
5) String; “ATAGGTACTT”, index: 248
6) String; “ATTTTCTATC”, index: 469
7) String; “AGCCGACATT”, index: 392
8) String; “ACCGTCCAGT”, index: 115
9) String; “GCATGCTATT”, index: 460
10) String; “CCCGGTTATA”, index: 53

We used the same input file for all algorithms. prepare_input.py does everything that we mentioned above.

Median String Algorithm

Median String algorithm is a brute force algorithm which means it looks every possible solution for the problem. Main idea of this algorithm is calculating hamming distance of all possible combinations of k-mers and calculating distances of them. The k-mer which has lowest distance will be the best k-mer to find it.

First, we run it for k=9.

```
AAAAAAATT
AAAAAATGT
AAAACCTGT
AAATACAAA
AAGCCCGAT
TATCACTTG
GGCCCCACT
Time:  1243.9030620840495
```

The execution time took 20 minutes. As can be seen from the results, it finds a better result during execution and replaces it with the old result. Our latest result; “GGCCCCACT”.

Secondly, we run it for k=10.

```
AAAAAAAAAT
AAAAAAAACT
AAAAAAATGT
AAAAAATGTA
AAAAATTCAG
AAAAATGCTG
AAAAGTGTGA
AATATGCCGG
AACGTGCAAA
ATTTCCATGA
TTTGCATGAG
Time:  5326.4866317919805
```

The execution time took 1 hour and 28 minutes. Again, the results show improvement as the program runs. Final result is; “TTTGCATGAG”.

Finally, we run it for k=11.

```
AAAAAAAAAAT
AAAAAAAAATT
AAAAATAAATT
AAAAATACAAA
AAAAACAAATT
AAAAAGTGTGA
AAATATCGTCC
AAATACAAAGT
TCAGCAATGTT
Time:  22579.043663583056
```

The execution time took 6 hours and 16 minutes.

The more k-mers we look at, the higher the execution time because the algorithm had to search for every possible k-mer from AA..AA to TT..TT. Since we added mutations to the string we were looking for at the beginning, the algorithm cannot find the searched string. Instead, it finds the k-mer which has lowest distance.

Randomized Motif Search

For this algorithm, we get one 10-mer motifs for every 10 lines from our input file. This 10-mer motifs selected as randomly. We find how many times the "A, C, G, T" nucleotides repeat in each line in the motifs we have chosen. In this way, we create profiles of motifs. We look at the 10-mer strings one by one for all the lines in the input file. We start at the beginning of the line and look at the first string, then move one step to the right and look at the other strings. We find a score for the strings we are looking at, using the values we found in the profile. Our aim is still finding the best score. Once we find the best score for the line, we replace it with the old motifs. So, our motifs are changed according to the scores. After changing the motifs, we calculate the profile again and find the new scores. we replace it them with the old ones again. The application continue until score can no longer be improved.

For the randomized motif search algorithm, we tried each 9,10,11-mers 5 times.

Firstly, we used the value k=9

Randomized k=9 1. Run time: 0.015 seconds

```
BEST MOTIFS:  ['CCGATTTAT', 'TCGTTTCAG', 'CCGAATCAG', 'TCGAATTAA', 'TCGTATCCG', 'TCGTTTTAG', 'TCGACTTGG', 'CCCTCTTAC', 'CCGAATCAA', 'CCGATTTCA']
SCORE:  28
CONSENSUS:  TCGAATTAG
Iteration number:  4
Time:  0.015016874996945262
```

Randomized k=9 2. Run time: 0.010 seconds

```
BEST MOTIFS:  ['TCCGATTTA', 'TGCGATACA', 'CGCAACACA', 'TTCGACACC', 'TGAGATATC', 'GGAAATACA', 'CCCGATATC', 'CGAGATTTA', 'TGCGGTACA', 'TGCAATTTC']
SCORE:  28
CONSENSUS:  TGCGATATA
Iteration number:  2
Time:  0.010356459009926766
```

Randomized k=9 3. Run time: 0.010 seconds

```
BEST MOTIFS:  ['CCGGATAAT', 'AGTGTTAGG', 'CCGAATCAG', 'GCGGCTGGG', 'CGGTATAGG', 'CCTACTCGT', 'AGTGATAGG', 'CGTGCTAAG', 'CCGGTTCGG', 'CGGGTTCGT']
SCORE:  32
CONSENSUS:  CCGGATAGG
Iteration number:  2
Time:  0.01069837500108406
```

Randomized k=9 4. Run time: 0.011 seconds

```
BEST MOTIFS:  ['AAGATGTCC', 'AGCTCTACT', 'AAGACATCA', 'AGCTCTACA', 'AGCATAACC', 'AAGACGTCA', 'AACTTGTTC', 'AAGATTTCA', 'AGCATGACA', 'ATGTTATCG']
SCORE:  34
CONSENSUS:  AACATGTCA
Iteration number:  3
Time:  0.011643167003057897
```

Randomized k=9 5. Run time: 0.007 seconds

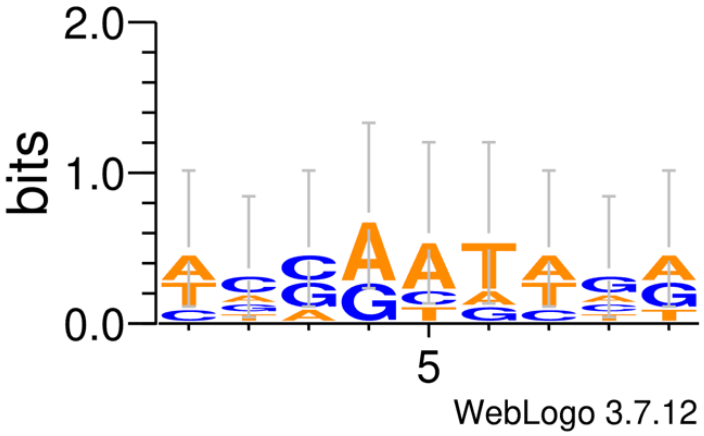
```
BEST MOTIFS:  ['TTTACATCT', 'AGCATACGT', 'TAAATACGC', 'ATAATAGAT', 'AGCACACGT', 'ATAACACGT', 'AAAACCTGT', 'TCCACACAT', 'ATAACAGGT', 'ATTACCTGT']
SCORE:  27
CONSENSUS:  ATAACACGT
Iteration number:  1
Time:  0.007201166998129338
```

Best Score = 27 Average Score= 29,8

Average iteration numbers for the randomized motif search when k=9; 2.4 iteration.

Average execution time for the randomized motif search when k=9; 0.016 seconds.

The motif logo for the consensus string for the 9-mer strings in the randomized motif search (for best result).



Secondly, we used the value k=10

Randomized k=10 1. Run time: 0.008 seconds

```
BEST MOTIFS:  ['TGAAGTCCGA', 'TGATTGCCGT', 'TGCAGTCCGC', 'TCCGAGCTCT', 'TGATTTCGCT', 'TGCGCTCTGT', 'TCAAGGCCGA', 'TGTAGGCTGA', 'TGCATGCGGT', 'TGATTGCCTG'
SCORE:  35
CONSENSUS:  TGAATGCCGT
Iteration number:  1
Time:  0.008009875018615276
```

Randomized k=10 2. Run time: 0.012 seconds

```
BEST MOTIFS:  ['GGTATTAGAA', 'GCTAATAGCA', 'CGAATTAGCG', 'GATAATAGAT', 'CCGAATAGCA', 'AGGAATAACC', 'GGGATTAACC', 'GGTAGCAGAA', 'CGTGATAGAA', 'AGTGTCAGCA'
SCORE:  31
CONSENSUS:  GGTAATAGCA
Iteration number:  2
Time:  0.012427208013832569
```

Randomized k=10 3. Run time: 0.012 seconds

```
BEST MOTIFS:  ['ATTCTCGTGA', 'AAATTCGTCA', 'AGTATTGTGA', 'TTCATAGTGC', 'TGACTGGTGC', 'TGTCTCGTGT', 'ATAATGTTTT', 'TGAATAGAGT', 'ATATTTGAGA', 'TTCATCGTGC'
SCORE:  37
CONSENSUS:  ATAATCGTGA
Iteration number:  2
Time:  0.012324290990363806
```

Randomized k=10 4. Run time: 0.009 seconds

```
BEST MOTIFS:  ['TGAGGATCTT', 'TGTCCATACG', 'TGTCCATGGG', 'TTTGCATCAG', 'TGTCGGAGTG', 'TTTCGCTGCG', 'TTACCAAAAG', 'TGCCGATAAG', 'TGACAACGAG', 'TGGCGCAGAG']
SCORE:  33
CONSENSUS:  TGTCGATGAG
Iteration number:  2
Time:  0.009227083006408066
```

Randomized k=10 5. Run time: 0.012 seconds

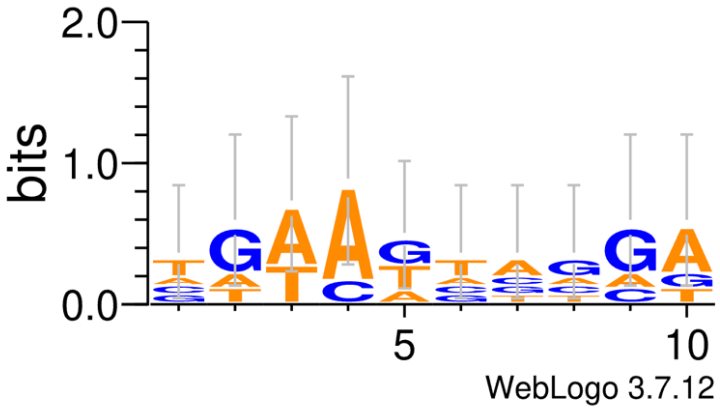
```
BEST MOTIFS:  ['TGAAGTCCGA', 'CAAACTAAGA', 'CCTACTAATA', 'TTAATTGAGA', 'CCATGTGAGA', 'CAAACTAGAA', 'TCAAGTCGGA', 'TAGAGTAGGA', 'CCGAATCAAA', 'CAAAGTAAGC']
SCORE:  32
CONSENSUS:  CAAAGTAAGA
Iteration number:  3
Time:  0.01276474999031052
```

Best Score = 31 Average Score= 33,6

Average iteration numbers for the randomized motif search when k=10; 2 iteration.

Average execution time for the randomized motif search when k=10; 0.016 seconds.

The motif logo for the consensus string for the 10-mer strings in the randomized motif search (for best result).



Thirdly, we used the value k=11

Randomized k=11 1. Run time: 0.006 seconds

```
BEST MOTIFS:  ['AGATGGACGAC', 'AGACTTACGCC', 'ACGTTTCACCA', 'AGATTGACCTA', 'GAATAGCACAC', 'AGATTGCCGAG', 'ATCTGGCCCCA', 'GGACGGCACCA', 'ATATTTGAGAA', 'TTATAGACCCC']
SCORE:  43
CONSENSUS:  AGATTGCCCCA
Iteration number:  1
Time:  0.006711959023959935
```

Randomized k=11 2. Run time: 0.009 seconds

```
BEST MOTIFS:  ['TGCTCAGTTAC', 'CGAGAGGTCTC', 'AGCGTTGTCAG', 'CGCCCAGTCAC', 'TGCAAAGACAG', 'TTCGCCGTCCT', 'TGCGCCGTTTG', 'CGCGGGGTCCC', 'AGCGCGGCCAC', 'AGCAATGTTAT']
SCORE:  38
CONSENSUS:  TGCGCAGTCAC
Iteration number:  2
Time:  0.009998624969739467
```

Randomized k=11 3. Run time: 0.011 seconds

```
BEST MOTIFS:  ['AGGATCTTGGT', 'AGCATCTCGCT', 'GGGAGCTTGGA', 'TGCAGCAAGCT', 'ATGATTTCGCT', 'GGCATTCTACT', 'GAGAATCTGCT', 'AGGAATTATGT', 'AGGATCACAGA', 'GGGACCCCGGT']
SCORE:  40
CONSENSUS:  AGGATCTTGCT
Iteration number:  2
Time:  0.011096582980826497
```

Randomized k=11 4. Run time: 0.007 seconds

```
BEST MOTIFS:  ['TTTCGGGGGTC', 'TTCCACGACTA', 'TGTCTCTAGTC', 'GTCCTATCCGA', 'GGGCACTTGTA', 'TGTCTCGTGTA', 'TCCCTATGCGT', 'GTGCACGTGTA', 'GGCCACGAGGA', 'TTGCATTTGTA']
SCORE:  44
CONSENSUS:  TTCCACTTGTA
Iteration number:  1
Time:  0.007502583030145615
```

Randomized k=11 5. Run time: 0.009 seconds

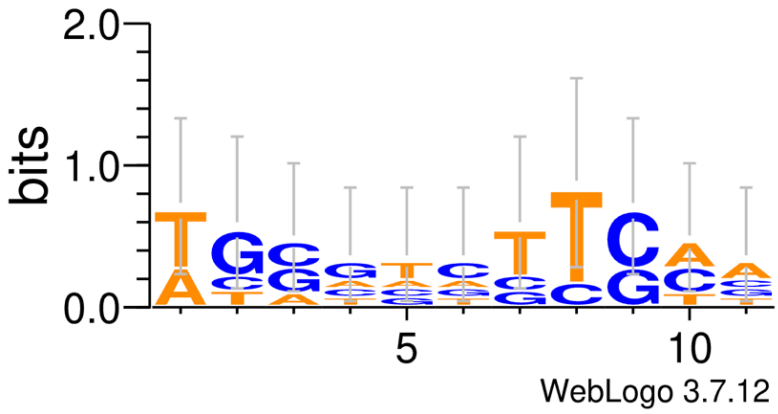
```
BEST MOTIFS:  ['TCGGGACTTAG', 'TCTCGTTTCAG', 'CGGCGAATTAG', 'TGGGGCTTCCC', 'CCTCGTATCCG', 'AAGGGTTTCGC', 'AATGTTTTTCC', 'CAGAATATCCG', 'TCTGGATTCAG', 'CCGATTATTGG']
SCORE:  45
CONSENSUS:  TCGGGTTTCAG
Iteration number:  1
Time:  0.009381208976265043
```

Best Score = 38 Average Score= 42

Average iteration numbers for the randomized motif search when k=11; 1.4 iteration.

Average execution time for the randomized motif search when k=11; 0.008 seconds.

The motif logo for the consensus string for the 11-mer strings in the randomized motif search (for best result).



Randomized Motif Search algorithm runs very fast, but not gives very good results in terms of finding best motifs comparing to Gibbs Sampler. Comparison and detailed information were given under ‘Conclusion’ section at the end of the report.

Gibbs Sampling

Gibbs Sampling algorithm works similar to the randomized motif search algorithm. In randomized motif search we were applying a change for each row, in Gibbs Sampling, we just select a random row and change it. First, we select 10 k-mer for each line with randomly, then we delete one of the lines in a random way. We create a profile with the remaining 9 lines. We add +1 to all profiles to give a chance to strings that have a probability of 0. We scan the 500-character line from which we took the k-mer we deleted, just to find the k-mer that will be selected instead of the k-mer we deleted. Then we throw biased die according to the score of k-mers in the scanned line. (The highest score is more likely to come). As a result of biased die, we place the new k-mer to selected line to be deleted and calculate the new score. Gibbs Sampler continue until the score of the algorithms no longer improve for given iteration number.

For the Gibbs Sampling algorithm, we tried each 9,10,11-mers 5 times. Every 50 iteration, algorithm checks whether score is improved or not in last 100 iterations. If score is not improved, then algorithm stops.

Firstly, we used the value k=9

Gibbs k=9 1. Run time: 0.248 seconds

```
BEST MOTIFS:  ['GTGACGTTT', 'TTGCCGTTA', 'GAGACTCTA', 'TTGACCTAA', 'CACACGTTA', 'ATGTCGCTT', 'GCGCCGTTT', 'GAGACAATA', 'GTGACTTTG', 'TCGCCGTTA']
SCORE:  27
CONSENSUS:  GTGACGTTA
Iteration number:  250
Time:  0.24849004100542516
```

Gibbs k=9 2. Run time: 0.401 seconds

```
BEST MOTIFS:  ['GGAAGCATT', 'TGAAGCATA', 'TGAAACATG', 'TGCAGCAAG', 'GGAAGCCTT', 'TGAAACCTC', 'AGAATAATG', 'TGAAACAGG', 'GGAAGCAGG', 'GGAAGCATG']
SCORE:  20
CONSENSUS:  TGAAGCATG
Iteration number:  400
Time:  0.40179195802193135
```

Gibbs k=9 3. Run time: 0.148 seconds

```
BEST MOTIFS:  ['GTGTCGTCT', 'CATACGTCT', 'CTCTAGTCT', 'CTTTCGTCC', 'TTGTGGTGG', 'GTCTCGTGT', 'CTCTCCACC', 'TTTCCGTCT', 'ATTTCCTCT', 'GGTTCGTGG']
SCORE:  27
CONSENSUS:  CTTTCGTCT
Iteration number:  150
Time:  0.1486006670165807
```

Gibbs k=9 4. Run time: 0.501 seconds

```
BEST MOTIFS:  ['CTAAGTGCC', 'CTTGTCGGG', 'CTACTCGCA', 'ATTGGGGCT', 'CTTGGCGGC', 'CTTGGTGCT', 'CTAGGCGCA', 'CTCCGCGGG', 'CTCGGCGTG', 'ATTGGAGCG']
SCORE:  26
CONSENSUS:  CTTGGCGCG
Iteration number:  500
Time:  0.5014766670065001
```

Gibbs k=9 5. Run time: 0.149 seconds

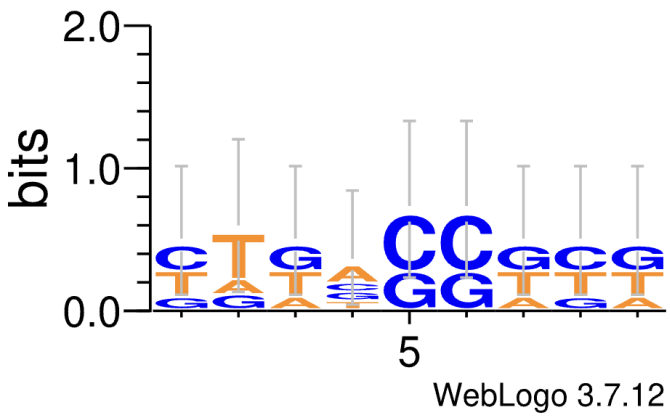
```
BEST MOTIFS:  ['GTGCACAAT', 'CCGACCGGC', 'TAGCCCCGT', 'TCGCCCAGT', 'AAGCCCGGT', 'CAACGAGGT', 'TAGCTGGCT', 'CCGCGGGGT', 'TAGACCGGT', 'TAAACCGAT']
SCORE:  28
CONSENSUS:  TAGCCCGGT
Iteration number:  150
Time:  0.149581459001638
```

Best Score = 20 Average Score= 25,6

Average iteration numbers for the gibbs sampling when k=9; 290 iteration.

Average execution time for the gibbs sampling when k=9; 0.289 seconds.

The motif logo for the consensus string for the 9-mer strings in the gibbs sampling (for best result).



Secondly, we used the value k=10

Gibbs k=10 1. Run time: 0.156 seconds

```
BEST MOTIFS:  ['ATAGTTCCGT', 'AGGTGACCGA', 'GGGTTATTGG', 'CGATAATAGA', 'AAGGAATTGC', 'CAATTATCGG', 'CAGTTTTTTC', 'TTGTTATCGC', 'ATTTTGTGGG', 'ATGTTATCGG']
SCORE:  37
CONSENSUS:  ATGTTATCGG
Iteration number:  150
Time:  0.15635354100959376
```

Gibbs k=10 2. Run time: 0.212 seconds

```
BEST MOTIFS:  ['AGGCAGACCA', 'ATCCCGCTCT', 'ATCTAGCCCC', 'AGCAAGCTCC', 'ATCCAGAGCG', 'AACTAGAACG', 'AGCTGGCTAC', 'AGCCCGATCC', 'AGTCCGCACT', 'AGTCAGCTCG']
SCORE:  31
CONSENSUS:  AGCCAGCTCC
Iteration number:  200
Time:  0.21288545802235603
```

Gibbs k=10 3. Run time: 0.207 seconds

```
BEST MOTIFS:  ['CTTTCTATCC', 'CTCTATCGAC', 'TTCTATCTCT', 'CTATCCGAGC', 'CTGTCTAGCT', 'TTATCGGCCA', 'GTATCTGGCC', 'CCGTCTGGGC', 'CTATCTGGGT', 'CTTTAGCTCC']
SCORE:  34
CONSENSUS:  CTATCTGGCC
Iteration number:  200
Time:  0.20736512501025572
```

Gibbs k=10 4. Run time: 0.307 seconds

```
BEST MOTIFS:  ['AAAGAACTAA', 'AAATTCGTCA', 'AAATACGCCA', 'ATTGACCTAA', 'ATAACCGACA', 'AAAGACGTCA', 'AAGTGCGCCG', 'AATTATGTAA', 'AAATTCGGAA', 'AATTTCGCAA']
SCORE:  30
CONSENSUS:  AAATACGTAA
Iteration number:  300
Time:  0.30763041600584984
```

Gibbs k=10 5. Run time: 0.205 seconds

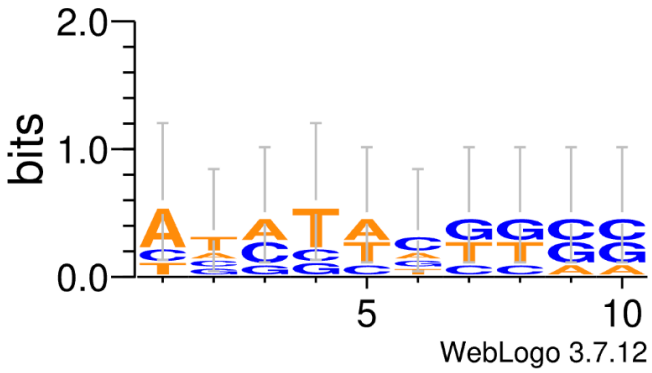
```
BEST MOTIFS:  ['TCCGTCTATG', 'TCGGGCTGGA', 'TCCTGTTGGC', 'TGCGGCTGGG', 'TCCCTCTCGC', 'GGCTTCTCGC', 'TACCTCAAGG', 'TCCGTCTGGG', 'TCGTTCAGCG', 'TGGCGCAGAG']
SCORE:  33
CONSENSUS:  TCCGTCTGGG
Iteration number:  200
Time:  0.20505162503104657
```

Best Score = 30 Average Score= 33

Average iteration numbers for the gibbs sampling when k=10; 210 iteration.

Average execution time for the gibbs sampling when k=10; 0.217 seconds.

The motif logo for the consensus string for the 10-mer strings in the gibbs sampling (for best result).



Thirdly, we used the value k=11

Gibbs k=11 1. Run time: 0.160 seconds

```
BEST MOTIFS:  ['TCTGCACTCGA', 'TTGCCGTTACA', 'TCCTCGCTCGA', 'TCGCCTCAAGA', 'TCGCTGTCTAG', 'TCCGCACTTGG', 'CCAGCGTAAGA', 'TTTCCATGAAA', 'TCGGCGTGATA', 'TTTGCGTTTCA']
SCORE:  41
CONSENSUS:  TCGGCGTTAGA
Iteration number:  150
Time:  0.16043279197765514
```

Gibbs k=11 2. Run time: 0.270 seconds

```
BEST MOTIFS:  ['AAGAACTAAGT', 'AAGACCCAAGC', 'TAGAGATAAGG', 'AATGTATATGG', 'AGGCTTAAATG', 'GAGTTTTAAGG', 'CCTCCCTATGC', 'GAGATTTAAGG', 'GAAAGTCATGG', 'AATGTTTAACG']
SCORE:  37
CONSENSUS:  AAGATTTAAGG
Iteration number:  250
Time:  0.270670166995842
```

Gibbs k=11 3. Run time: 0.268 seconds

```
BEST MOTIFS:  ['ACAATAAAGAA', 'ACGAGAAACGT', 'GAGATAAGGAC', 'GCGATAATAGA', 'ACGACATACAA', 'ACGTCACAGAA', 'AGAATAATGTT', 'GTTATAAAATG', 'AGGATCACAGA', 'ACGCCAAAGTA']
SCORE:  38
CONSENSUS:  ACGATAAAGAA
Iteration number:  250
Time:  0.26875612494768575
```

Gibbs k=11 4. Run time: 0.267 seconds

```
BEST MOTIFS:  ['AGGCACTTTCT', 'ACTCGCGTTCG', 'GTCCTGTTTCG', 'AGTCGTTGACG', 'CGTTACTTTCC', 'TGTCGCTTTTG', 'CGTTTGTGACG', 'CGGTAGTGTCG', 'AGAAGCTTTCT', 'CTTTGCGTTTC']
SCORE:  40
CONSENSUS:  AGTCGCTTTCG
Iteration number:  250
Time:  0.2676890000024587
```

Gibbs k=11 5. Run time: 0.271 seconds

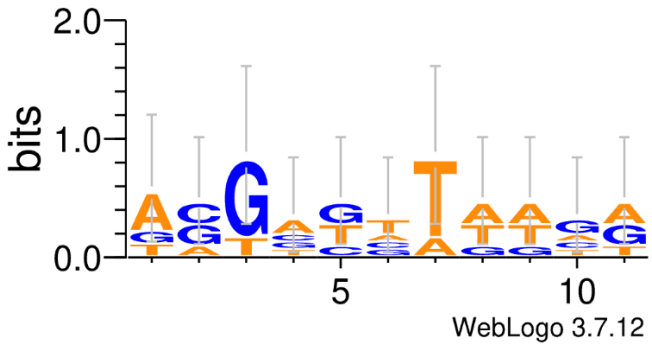
```
BEST MOTIFS:  ['GGATCTTGGTG', 'CGGTGTTCTTG', 'AGGTCCTGTTT', 'GGGTGTCCTAT', 'GGGTTTTGAAT', 'GTGTCTCGTGT', 'AGGAGTTGTGC', 'GAGTGTTGTTA', 'TGGTGTTCCGC', 'GTGTGTAGCAG']
SCORE:  36
CONSENSUS:  GGGTGTTGTTT
Iteration number:  250
Time:  0.27189058397198096
```

Best Score = 36 Average Score= 38,4

Average iteration numbers for the gibbs sampling when k=11; 230 iteration.

Average execution time for the gibbs sampling when k=11; 0.247 seconds.

The motif logo for the consensus string for the 11-mer strings in the gibbs sampling.



The key of this algorithm in terms of finding good result (lower score) is iteration number. When iteration number is increased, algorithm searches and finds better motifs. The disadvantage of increasing iteration number is increasing execution time.

Conclusion

The algorithm that gives the best result among all the algorithms is the median string algorithm. Because while others are proceeding in a random way, the median string algorithm is looking at it by examining it one by one. The worst feature of the median string is that the execution time takes too long. That is why this algorithm is not preferred to used usually.

The randomized motif search algorithm can find the result by at least iterations. in this way, the execution time is shorter than the others. If iteration number is increased it can give better results. The reason it doesn't give the best results is because it randomly selects all motifs. Accordingly, when the value of any nucleotide in the profile is 0, there is no possibility of choosing the motif in which that nucleotide is located.

While each line is changed in Randomized motif search, it is changed as a single line in Gibbs Sampling. Since it looks at each line individually, the number of iterations increases. The Gibbs sampler algorithm takes longer than randomized motif search because there are many more iterations. Since it looks line by line, it finds better results as the number of iterations increases.