Report on BioInformatics Assignment

CSE463

Mashroor Hasan Bhuiyan(1905069)
Md. Nafiu Rahman (1905077)
Kazi Reyazul Hasan(1905082)
Wasif Jalal (1905084)
Mubasshira Musarrat(1905088)

Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology (BUET)

March 14, 2024

Contents

Chapter 1

Data

In order to evaluate the performance of our selected methods, modifications, and third-party tools, they were applied to three distinct datasets derived from the TRANSFAC database (Matys, 2003), as done in the provided publication (Karaboga and Aslan, 2018).

1.1 Datasets Overview

The first dataset, labeled as **hm03r**, consists of 10 sequences from humans, each 1500 nucleotides in length. The complexity and diversity of the human genome necessitate sophisticated approaches to accurately identify motifs. The second and third datasets, **yst04r** and **yst08r**, are derived from yeast, containing 7 and 11 sequences respectively, each 1000 nucleotides long. The compact nature of yeast genomes presents a different set of challenges, including dense packing of regulatory elements.

1.2 Biomarker/Ground Truth

Exact ground truth (motifs) for selected datasets cannot be found; however, it is possible to examine identified motifs and compare them with known motifs. For this purpose, the Tomtom motif comparison tool of the MEME Suite was used (Gupta et al., 2007).

Chapter 2

Methods

2.1 Gibbs Sampler

Gibbs Sampler is a widely used algorithm for motif discovery in biological sequences. In our study, we applied the Gibbs Sampler algorithm to identify conserved motifs within a set of DNA sequences. The algorithm works by iteratively sampling potential motif occurrences from the input sequences and updating the motif model based on the sampled occurrences. Here's an overview of the steps involved:

Algorithm 1: Gibbs Sampler Algorithm

Input: Sequences S, Motif length k, Number of sequences t, Number of iterations N

Output: Best motifs M, Best score *score*, Consensus sequence *consensus*

Initialize M by randomly selecting k-mers from S;
$best\_motifs \leftarrow M$;
$best\_score, consensus \leftarrow$ Score($best\_motifs$, k);
for $j \leftarrow 1$ **to** N **do**
    Choose a random sequence index i from 1 to t;
    Create profile matrix P excluding motif $M[i]$;
    Sample a new motif for sequence $S[i]$ using profile-weighted random selection;
    Calculate score *current_score* and consensus for the updated motifs;
    if *current_score* $<$ *best_score* **then**
        $best\_motifs \leftarrow M$;
        best_score $\leftarrow$ *current_score*;
    end
end
return *best_motifs, best_score, consensus*;

We implemented the Gibbs Sampler algorithm using Python and tuned its parameters to achieve optimal motif discovery performance.

2.2 Randomized Motif Search

Randomized Motif Search is another popular algorithm for motif discovery, particularly useful for its simplicity and effectiveness. In our study, we employed the Randomized Motif Search algorithm to identify conserved motifs across a set of DNA sequences. The algorithm involves the following steps:

Algorithm 2: Randomized Motif Search Algorithm

Input: Sequences S, Motif length k, Number of sequences t, Number of iterations N

Output: Best motifs M, Best score *score*, Consensus sequence *consensus*

Initialize M by randomly selecting k-mers from S;
$best\_motifs \leftarrow M$;
$best\_score, consensus \leftarrow \text{Score}(best\_motifs, k)$;
while *True* **do**
    Create profile matrix P based on M;
    Update M by choosing the highest probability k-mer from each sequence using P;
    Calculate score *current_score* for the updated motifs;
    if *current_score* $<$ *best_score* **then**
        $best\_motifs \leftarrow M$;
        best_score $\leftarrow$ *current_score*;
    end
    else
        Break;
    end
end
return *best_motifs, best_score, consensus*;

We implemented the Randomized Motif Search algorithm and adjusted its parameters to optimize motif discovery performance for our specific dataset.

2.3 Enhanced Randomized Motif Search

We build upon the traditional Randomized Motif Search approach by incorporating multiple iterations of the entire search process to more effectively navigate the solution space. This enhancement aims to mitigate the risk of converging on local optima by initiating the search from various random starting points, thereby increasing the probability of identifying the globally optimal motif set. Herein, we mention the steps of this augmented algorithm:

Algorithm 3: Enhanced Randomized Motif Search Algorithm

Input: Sequences S, Motif length k, Number of sequences t, Total iterations T

Output: Best motifs M, Best score *score*, Consensus sequence *consensus*

Initialize $global\_best\_motifs$ with an arbitrary high score;
Initialize *global_best_score* to infinity;
for $iteration \leftarrow 1$ **to** T **do**
    Seed the random number generator with the current time;
    Randomly select initial k-mers from S to form M;
    $best\_motifs \leftarrow M$;
    $best\_score, consensus \leftarrow$ Score($best\_motifs$, k);
    while *True* **do**
        Create profile matrix P based on M;
        Update M by choosing the highest probability k-mer from each sequence using P;
        Calculate score *current_score* for the updated motifs;
        if *current_score* $<$ *best_score* **then**
            $best\_motifs \leftarrow M$;
            best_score $\leftarrow$ *current_score*;
        end
        else
            Break;
        end
    end
    if *best_score* $<$ *global_best_score* **then**
        $global\_best\_motifs \leftarrow best\_motifs$;
        global_best_score $\leftarrow$ *best_score*;
    end
end
return $global\_best\_motifs$, *global_best_score*, *Consensus(*$global\_best\_motifs$*)*;

The dynamic seeding mechanism further ensures a varied and comprehensive exploration in each run, enhancing the overall efficiency of the motif finding process.

2.4 Modified Gibbs Sampler with Exempted Sequences

The Modified Gibbs Sampler is an extension of the traditional Gibbs Sampler algorithm. This variant introduces the concept of exempted sequences, allowing a predefined number of sequences to remain unaltered during the motif sampling process. This modification aims to explore the impact of stabilizing certain sequences on the overall motif discovery process. Below we detail the steps of the modified algorithm:

Algorithm 4: Modified Gibbs Sampler Algorithm with Exempted Sequences

Input: Sequences S, Motif length k, Number of sequences t, Number of iterations N, Number of exempt sequences *num_exempt*

Output: Best motifs M, Best score *score*, Consensus sequence *consensus*

Initialize M by randomly selecting k-mers from S;
$best\_motifs \leftarrow M$;
$best\_score, consensus \leftarrow$ Score($best\_motifs$, k);
for $j \leftarrow 1$ **to** N **do**
    exempt_indices $\leftarrow$ Randomly select *num_exempt* indices from 1 to t;
    for $i \leftarrow 1$ **to** t **do**
        if *i not in exempt_indices* **then**
            Create profile matrix P excluding motifs in *exempt_indices*;
            Sample a new motif for sequence $S[i]$ using profile-weighted random selection based on P;
            Update motif $M[i]$ with the sampled motif;
        end
    end
    Calculate score *current_score* and consensus for M including all motifs;
    if *current_score* $<$ *best_score* **then**
        $best\_motifs \leftarrow M$;
        best_score $\leftarrow$ *current_score*;
    end
end
return $best\_motifs$, *best_score, consensus*;

We iteratively sample potential motif occurrences from a set of DNA sequences while fixing a subset of the motifs, chosen randomly at the start of each iteration, to explore the motif space more diversely. By exempting a fraction of sequences from updates, we aim to identify more accurate motifs in selected k-mers.

2.5 Targeted Gibbs Sampler with Score-Based Selection

The Targeted Gibbs Sampler is an innovative adaptation of the traditional Gibbs Sampler algorithm used for motif discovery in biological sequences. Unlike the standard approach, which randomly selects sequences for updating, this version prioritizes the update of sequences that, when temporarily excluded, most decrease the overall motif set score. Below, we outline the steps of this targeted algorithm:

Algorithm 5: Targeted Gibbs Sampler Algorithm with Score-Based Selection

Input: Sequences S, Motif length k, Number of sequences t, Number of iterations N

Output: Best motifs M, Best score *score*, Consensus sequence *consensus*

Initialize M by randomly selecting k-mers from S;
$best\_motifs \leftarrow M$;
$best\_score, consensus \leftarrow$ Score($best\_motifs$, k);
for $j \leftarrow 1$ **to** N **do**
    Identify the sequence $M[i]$ whose removal most decreases the score;
    Create profile matrix P excluding motif $M[i]$;
    Sample a new motif for sequence $S[i]$ using profile-weighted random selection based on P;
    Update motif $M[i]$ with the sampled motif;
    Recalculate score *current_score* and consensus for the updated motifs;
    if *current_score* $<$ *best_score* **then**
        $best\_motifs \leftarrow M$;
        best_score $\leftarrow$ *current_score*;
    end
end
return $best\_motifs$, *best_score, consensus*;

This modified algorithm introduces a more deliberate approach to motif refinement by focusing on the weakest links within the motif set.

Chapter 3

Software

We used MEME and STREME software tools from https://meme-suite.org/meme/tools/streme and https://meme-suite.org/meme/tools/meme. MEME discovers novel, ungapped motifs (recurring, fixed-length patterns) in sequences (sample output from sequences). MEME splits variable-length patterns into two or more separate motifs. STREME discovers ungapped motifs (recurring, fixed-length patterns) that are enriched in your sequences or relatively enriched in them compared to control sequences (sample output from sequences).

3.1 Commands to run

To run MEME type the following command -

```
meme <input file in fasta format> -dna -oc .  -nostatus -time 14400 -mod zoops
-nmotifs 10 -minw <min k> -maxw <max k> -objfun classic -revcomp -markov_order 0
```

To run STREME type the following command -

```
streme --verbosity 1 --oc .  --dna --totallength 4000000 --time 14400
--minw <min k> --maxw <max k> --thresh 0.05 --align center --p <input file in fasta
format>
```

3.2 Scripts to run

To run the tools run tools.sh . The shell script file is-

```
export PATH=\$HOME/meme/bin:\$HOME/meme/libexec/meme-:\$PATH
methods=("meme" "streme")
datasets=("hm03" "yst04r" "yst08r")

for d in \${datasets[@]}; do
    for m in \${methods[@]}; do
```

```
        for k in {8..15}; do
            mkdir -p "tool_output/\$d/\$m/k_\$k"
            cd "tool_output/\$d/\$m/k_\$k"
            echo -e "\nTool: \$m\nDataset: \$d\n k=\$k\n"
            if [ "\$m" = "meme" ]; then
                meme "../../../../\$d.txt" -dna -oc . -nostatus -time 14400
                -mod zoops -nmotifs 10 -minw \$k -maxw \$k -objfun classic -revcomp
                -markov_order 0
            elif [ "\$m" = "streme" ]; then
                streme --verbosity 1 --oc . --dna --totallength
                4000000 --time 14400 --minw \$k --maxw \$k
                --thresh 0.05 --align center  --p "../../../../\$d.txt"
            fi
            cd ../../../..
        done
    done
done
```

To calculate the scores run toolscore.sh . The shell script file is-

```
export PATH=$HOME/meme/bin:$HOME/meme/libexec/meme-:$PATH
methods=("meme" "streme")
datasets=("hm03" "yst04r" "yst08r")
for m in ${methods[@]}; do
    for d in ${datasets[@]}; do
        for k in {8..15}; do
            mkdir -p "tool_output/$d/$m/k_$k"
            cd "tool_output/$d/$m/k_$k"
            echo -e "\nTool: $m\nDataset: $d\n k=$k\n"
            if [ "$m" = "meme" ]; then
                python3 "../../../../meme_score.py" meme.txt
            elif [ "$m" = "streme" ]; then
                python3 "../../../../streme_score.py" sites.tsv
            fi
            cd ../../../..
        done
    done
done
```

Chapter 4

Results

4.1 Exp. configuration

The motif outputs from the two third-party tools MEME and STREME were parsed into motif matrices. The standard mismatch-scoring similar to Gibbs sampling and Randomized Motif Search was applied on the outputs of MEME and STREME to compare them with the methods we implemented.

4.2 Comparison

Table 4.1: Performance of Meme Tool on Various Datasets

Dataset	k	**Score**	**Consensus**
hm03	8	4	TGGACCCA
hm03	9	7	TCTGTCTCT
hm03	10	11	CTCTGTCTCT
hm03	11	14	GTCTCTGTCCC
hm03	12	17	TGTCTCTGTCCC
hm03	13	18	AATGAAAAAAAAA
hm03	14	26	GCTCTGACACTCAC
hm03	15	26	GAATGAAAAAAAAAT
yst04r	8	2	TTTCTGGC
yst04r	9	4	TTTTCTGGC
yst04r	10	8	CTTTTCTGGC
yst04r	11	7	TCTTTTCTTCC
yst04r	12	4	TTTTTTTTTCTT
yst04r	13	10	TTTTTTCTTCCTT
yst04r	14	8	TTTTTTTTCTTTTC
yst04r	15	9	TTTTTTTTTCTTTTC
yst08r	8	5	TTTCTGGC
yst08r	9	4	TTTCTTTTT
yst08r	10	7	GAAAAAAAAA
yst08r	11	6	TTTTTTTTTTT
yst08r	12	9	AAAAAAAAAAAT
yst08r	13	16	TTTTTTTTTTTCC
yst08r	14	20	AGGAAAAAAAAAAA
yst08r	15	23	TTTTTTTTTTTTTCC

Table 4.2: Performance of Streme Tool on Various Datasets

Dataset	k	**Score**	**Consensus**
hm03	8	8	AAACAAAG
hm03	9	10	AGACACAGA
hm03	10	12	AAAGAAGAAT
hm03	11	12	GAGACTCACAG
hm03	12	22	AAACAAAGYGAA
hm03	13	25	AAAACAAAGCGAA
hm03	14	24	GAATGAAAAMAAAA
hm03	15	32	AAAAAAAAAGTGAAR
yst04r	8	6	AATTAATT
yst04r	9	28	WRATTAATYW
yst04r	10	10	AAGGTATATA
yst04r	11	12	ACAAGAGAGAA
yst04r	12	15	AAAAAATTAAKR
yst04r	13	8	AGAAAAGAAAAAA
yst04r	14	18	ARAAAARAAAAATA
yst04r	15	12	AGAAAAGAAAAAAAA
yst08r	8	9	GTAAATAA
yst08r	9	8	AAGAAAAGA
yst08r	10	10	AGGAAGAAAA
yst08r	11	13	GAAAAGAAAAA
yst08r	12	8	AAAAAAAAAAAT
yst08r	13	11	AAAAAAAAAAATA
yst08r	14	18	AAAAAAAAAAATAG
yst08r	15	28	AAAAAAAAAAATAGN

Table 4.3: Performance of Gibbs Sampler on various Datasets

Dataset	k	**Score**	**Consensus**
hm03	8	9	AAAAAAAA
hm03	9	12	AAAATAAAA
hm03	10	15	AAAAAAATAA
hm03	11	19	AATGAAAAAAA
hm03	12	19	AAAAAAAATAAA
hm03	13	24	AAAAAAAATAAAA
hm03	14	26	AATGAAAAAAAAAT
hm03	15	28	AGAAAAAAAAATAAA
yst04r	8	2	TTTTTTTT
yst04r	9	6	AAAAAAAAA
yst04r	10	6	TTTTTTTTCT
yst04r	11	8	AAAAAAAAAAA
yst04r	12	9	AAAAAAAAAAAA
yst04r	13	10	TTATTTTTCTTTT
yst04r	14	14	AAAAAAAAAAAACA
yst04r	15	14	TTATTTTTCTTTTTT
yst08r	8	6	TTTTTTTT
yst08r	9	8	AAGAAAAAA
yst08r	10	9	TTTTTTTTTT
yst08r	11	12	ATTTTTTTTTT
yst08r	12	13	TATTTTTTTTTT
yst08r	13	21	AAAAAAAAAAAAA
yst08r	14	27	AAAAAAAAAAAAAA
yst08r	15	30	TTTTATTTTTTTTTT

Table 4.4: Performance of Modified Gibbs Sampler on various Datasets

Dataset	k	**Score**	**Consensus**
hm03	8	10	AAACAAAA
hm03	9	17	GAGAAAACA
hm03	10	25	ACTAAAACAC
hm03	11	28	AGTTTTCTTTC
hm03	12	23	AAGAAAAATCAA
hm03	13	29	AAATGGAAAAGAG
hm03	14	38	TAGATAAAAAAATA
hm03	15	45	TCCCTTGAGCCCAGG
yst04r	8	5	TCTTTCTT
yst04r	9	10	TTTTTTTTC
yst04r	10	16	TTTGCATGTA
yst04r	11	10	TTCTTTTTTTT
yst04r	12	14	CATATATAAATA
yst04r	13	18	AAGGAAAAAAAAA
yst04r	14	15	TTTATTTTTTTTTT
yst04r	15	24	TTTTTTCTTAAAACT
yst08r	8	14	AAAAATAA
yst08r	9	12	TATTTTTTT
yst08r	10	21	AAAATTATTT
yst08r	11	15	TTTATTTTTCT
yst08r	12	22	TTTTTTTTTCTT
yst08r	13	29	CAAAAAAAAAAAA
yst08r	14	37	ATTTTTCTTCTCCA
yst08r	15	37	ATGAAAAAAGAAAAA

Table 4.5: Performance of Enhanced Randomized-Motif on various Datasets

Dataset	k	**Score**	**Consensus**
hm03	8	6	CTCTGTCC
hm03	9	12	TCTCCTTCC
hm03	10	18	TGGAAGAGAG
hm03	11	19	ATGAAAAAAAA
hm03	12	25	ATGGAAAAGATA
hm03	13	27	AGAAAGAGAGAAA
hm03	14	30	AGAAAAAGAGAAAG
hm03	15	31	AGCCAACAAAATAAA
yst04r	8	1	ATTTTTTT
yst04r	9	5	ATTTTTTTT
yst04r	10	6	TTTTTTTTCT
yst04r	11	14	ATCCTTTTCTT
yst04r	12	12	AAAAAAAAAAAA
yst04r	13	14	ATTTTTTTTCTTT
yst04r	14	23	AAGAAGAAAAAAAA
yst04r	15	22	AAAAAAAAAAAAAAA
yst08r	8	4	ATTTTTTT
yst08r	9	6	ATTTTTTTT
yst08r	10	9	ATTTTTTTTT
yst08r	11	12	TATTTTTTTTT
yst08r	12	16	ATTTTTTTTTTT
yst08r	13	20	CTATTTTTTTTTT
yst08r	14	27	ATTTTTTTTTTTTT
yst08r	15	32	AAAAAAAAAAAAAAA

Table 4.6: Performance of Randomized-Motif on hm03 Dataset

k	**Score**	**Consensus**
8	5	AAAATAAA
9	9	CTCTGTCCC
10	12	GACACAGGGA
11	15	GACACAGGGAG
12	18	AAAAAAAATAAA
13	21	AAAAAAATAAAAA
14	24	AGCAAACAAAATAA
15	28	AGCAAACAAAATAAA

Table 4.7: Summary of Gibbs Sampler Exempt Results

Dataset	k	**Score**	**Consensus**
hm03	8	7	AAAATAAA
hm03	9	12	AAAAAAAAA
hm03	10	13	AAAATAAAAA
hm03	11	16	AAAAAAAATAA
hm03	12	17	AAAAAAAATAAA
hm03	13	21	AAAAAAAATAAAA
hm03	14	24	AATGAAAAAAAAAT
hm03	15	27	AGTAAACAAAATAAA
yst04r	8	3	TTTTTTTT
yst04r	9	4	ATTTTTTTT
yst04r	10	5	TTTTTTTTCT
yst04r	11	7	TATTTTTCTTT
yst04r	12	8	TATTTTTCTTTT
yst04r	13	9	TTATTTTTCTTTT
yst04r	14	14	AAAAAAAAAAAACA
yst04r	15	14	TTATTTTTCTTTTTT
yst08r	9	8	AAAAAAAAA
yst08r	10	9	TTTTTTTTTT
yst08r	11	13	AAAAAAAAAAA
yst08r	12	13	TATTTTTTTTTT
yst08r	13	17	TATTTTTTTTTTT
yst08r	14	23	TATTTTTTTTTTTT
yst08r	15	30	AAAAAAAAAAAAAAA

Chapter 5

Conclusion

Throughout this work, we have done a comprehensive exploration of motif discovery algorithms, with a particular focus on the Gibbs Sampler and Randomized Motif Search methodologies. Our journey included the implementation of standard approaches, innovative modifications to enhance their effectiveness, and the application of third-party tools like MEME and STREME for comparative analysis.

5.1 Insights Gained

The modifications to the Gibbs Sampler algorithm, designed to introduce a more targeted selection of sequences for updating and the incorporation of exempt sequences, have shown promising results. Similarly, the Enhanced Randomized Motif Search strategy, which emphasizes multiple iterations from varied initial conditions, highlighted the importance of exploring the motif space comprehensively. By avoiding premature convergence on suboptimal motifs, this approach underlines the stochastic nature of biological data analysis.

5.2 Challenges Encountered

One of the main challenges in motif discovery remains the validation of identified motifs due to the absence of a universally accepted "ground truth." The reliance on tools like Tomtom for comparison against known motifs databases underscores the difficulty in ensuring the biological relevance of newly discovered motifs.

5.3 Future Directions

Looking ahead, there are several paths for further research and development in the field of motif discovery. The integration of machine learning techniques, particularly deep learning, offers an exciting road for the development of algorithms that can learn complex patterns and predict motifs with higher accuracy.

References

Gupta, S., Stamatoyannopoulos, J. A., Bailey, T. L., and Noble, W. S. (2007). Quantifying similarity between motifs. *Genome biology*, 8:1–9.

Karaboga, D. and Aslan, S. (2018). Discovery of conserved regions in dna sequences by artificial bee colony (abc) algorithm based methods. *Natural Computing*, 18(2):333–350.

Matys, V. (2003). Transfac(r): transcriptional regulation, from patterns to profiles. *Nucleic Acids Research*, 31(1):374–378.