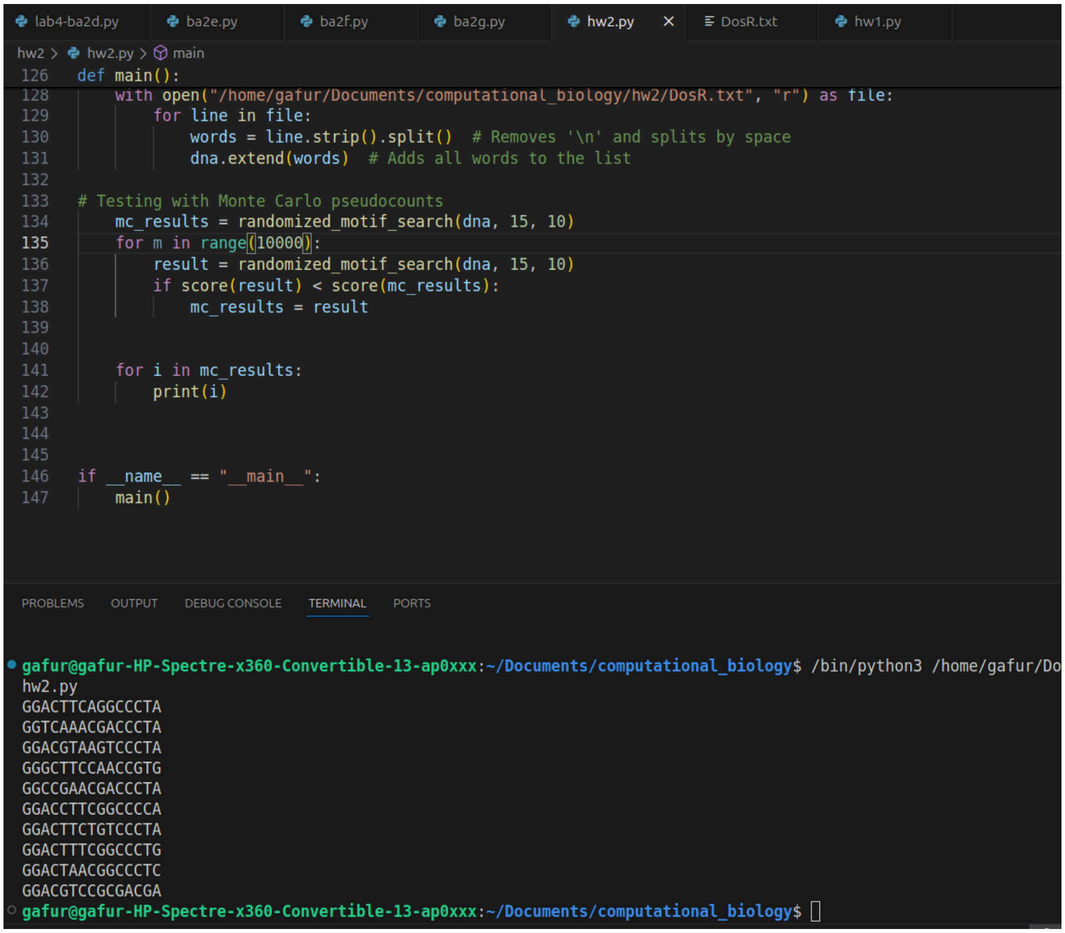
Motif Finding in "Upstream" Sequences

In my first trial, I attempted motif finding within the upstream sequences of ten selected genes. The initial sequences analyzed were:

Input Sequences:

1. GGACTTCAGGCCCTA
2. GGTCAAACGACCCTA
3. GGACGTAAGTCCCTA
4. GGGCTTCCAACCGTG
5. GGCCGAACGACCCTA
6. GGACCTTCGGCCCCA
7. GGACTTCTGTCCCTA
8. GGACTTTCGGCCCTG
9. GGACTAACGGCCCTC
10. GGACGTCCGCGACGA



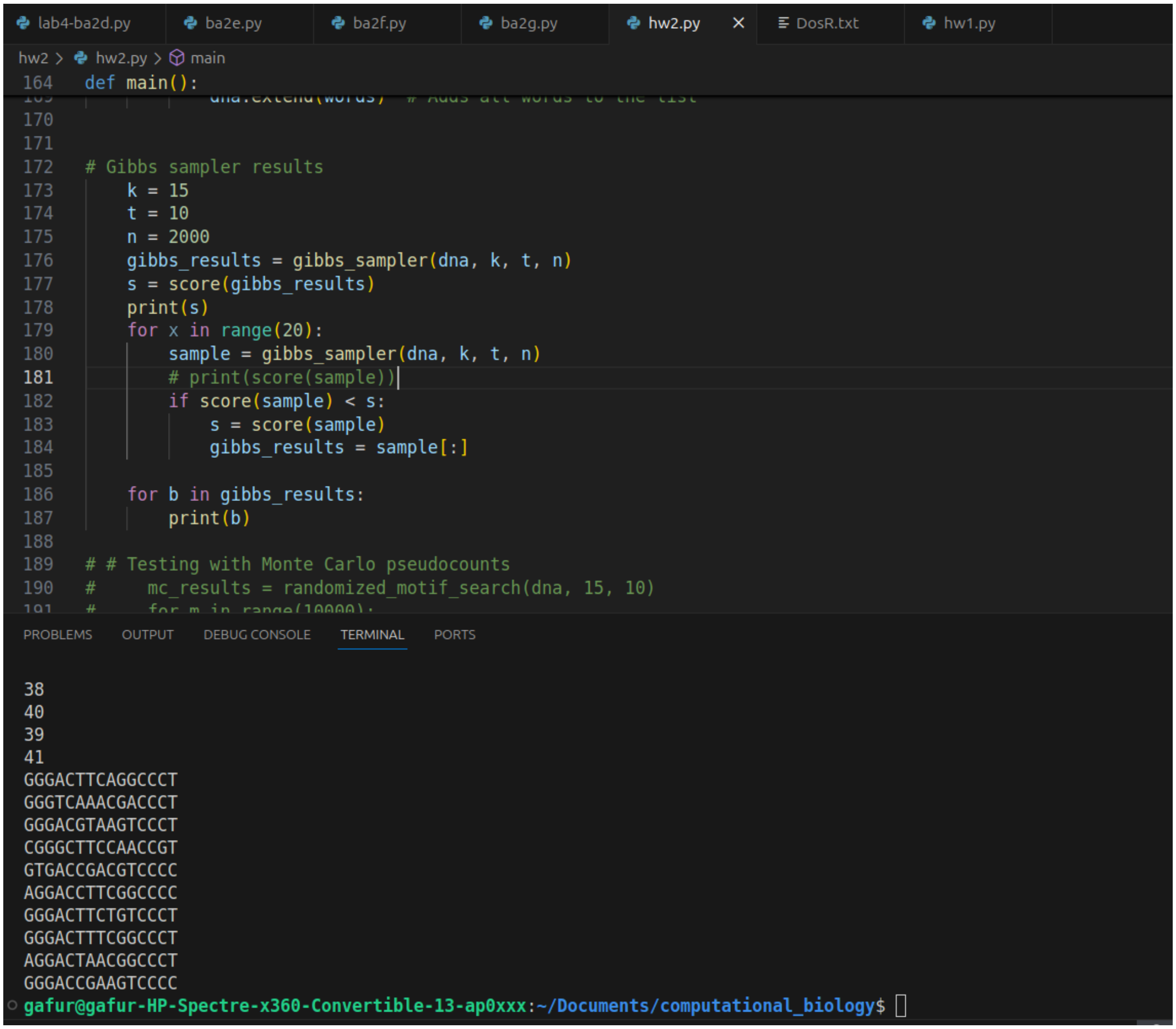
```python
def main():
    with open("/home/gafur/Documents/computational_biology/hw2/DosR.txt", "r") as file:
        for line in file:
            words = line.strip().split()  # Removes '\n' and splits by space
            dna.extend(words)  # Adds all words to the list

# Testing with Monte Carlo pseudocounts
    mc_results = randomized_motif_search(dna, 15, 10)
    for m in range(10000):
        result = randomized_motif_search(dna, 15, 10)
        if score(result) < score(mc_results):
            mc_results = result


    for i in mc_results:
        print(i)



if __name__ == "__main__":
    main()
```

```
gafur@gafur-HP-Spectre-x360-Convertible-13-ap0xxx:~/Documents/computational_biology$ /bin/python3 /home/gafur/Do
hw2.py
GGACTTCAGGCCCTA
GGTCAAACGACCCTA
GGACGTAAGTCCCTA
GGGCTTCCAACCGTG
GGCCGAACGACCCTA
GGACCTTCGGCCCCA
GGACTTCTGTCCCTA
GGACTTTCGGCCCTG
GGACTAACGGCCCTC
GGACGTCCGCGACGA
gafur@gafur-HP-Spectre-x360-Convertible-13-ap0xxx:~/Documents/computational_biology$ 
```

Results of Gibbs Sampling Algorithm

The Gibbs sampling algorithm extracted the following motifs:

1. GGGACTTCAGGCCCT
2. GGGTCAAACGACCCT
3. GGGACGTAAGTCCCT
4. CGGGCTTCCAACCGT
5. GTGACCGACGTCCCC
6. AGGACCTTCGGCCCC
7. GGGACTTCTGTCCCT
8. GGGACTTTCGGCCCT
9. AGGACTAACGGCCCT
10. GGGACCGAAGTCCCC



```python
def main():
            dna.extend(words)  # Adds all words to the list


# Gibbs sampler results
    k = 15
    t = 10
    n = 2000
    gibbs_results = gibbs_sampler(dna, k, t, n)
    s = score(gibbs_results)
    print(s)
    for x in range(20):
        sample = gibbs_sampler(dna, k, t, n)
        # print(score(sample))
        if score(sample) < s:
            s = score(sample)
            gibbs_results = sample[:]

    for b in gibbs_results:
        print(b)

# # Testing with Monte Carlo pseudocounts
#     mc_results = randomized_motif_search(dna, 15, 10)
#     for m in range(10000):
```

```
38
40
39
41
GGGACTTCAGGCCCT
GGGTCAAACGACCCT
GGGACGTAAGTCCCT
CGGGCTTCCAACCGT
GTGACCGACGTCCCC
AGGACCTTCGGCCCC
GGGACTTCTGTCCCT
GGGACTTTCGGCCCT
AGGACTAACGGCCCT
GGGACCGAAGTCCCC
gafur@gafur-HP-Spectre-x360-Convertible-13-ap0xxx:~/Documents/computational_biology$
```

Brute Force Approach

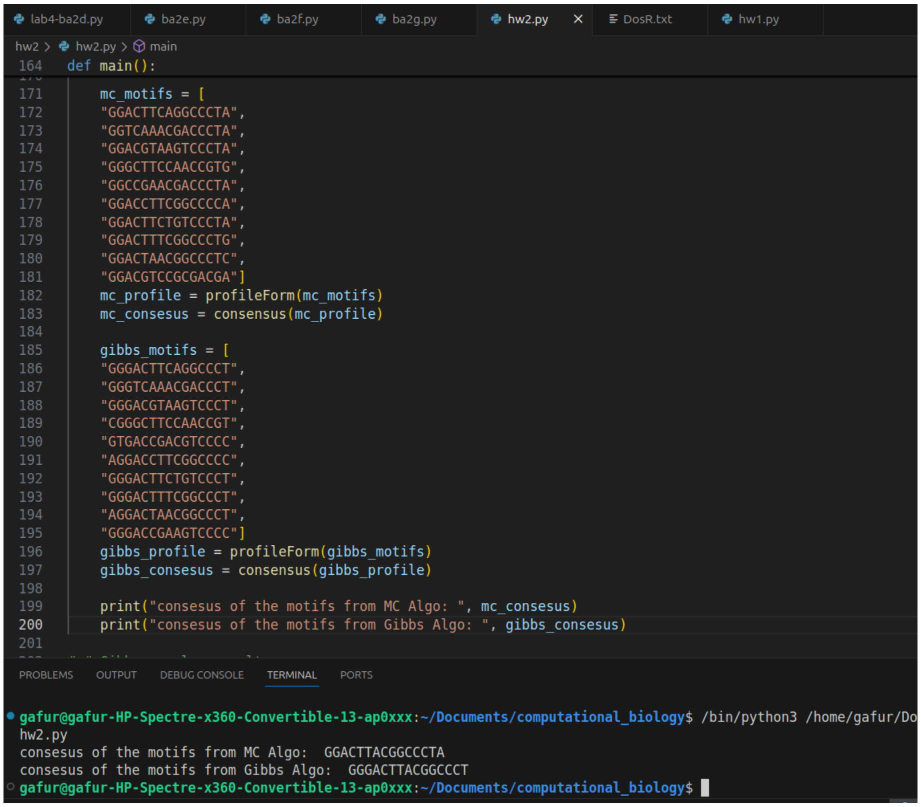
Before employing probabilistic algorithms, I attempted to find motifs using a brute-force approach. However, this method proved to be inefficient due to its computational complexity. The runtime increased exponentially, making it impractical for large-scale analysis. The difficulty in motif discovery using brute force underscores the necessity of heuristic or probabilistic approaches, such as Gibbs sampling and Markov Chain Monte Carlo (MCMC) methods.

Consensus Motifs

By applying different motif-finding algorithms, I obtained the following consensus motifs:

- **Markov Chain Algorithm Consensus: GGACTTACGGCCCTA**
- **Gibbs Sampling Algorithm Consensus: GGGACTTACGGCCCT**

These motifs suggest a recurring pattern in the upstream sequences, potentially indicating a conserved regulatory element.

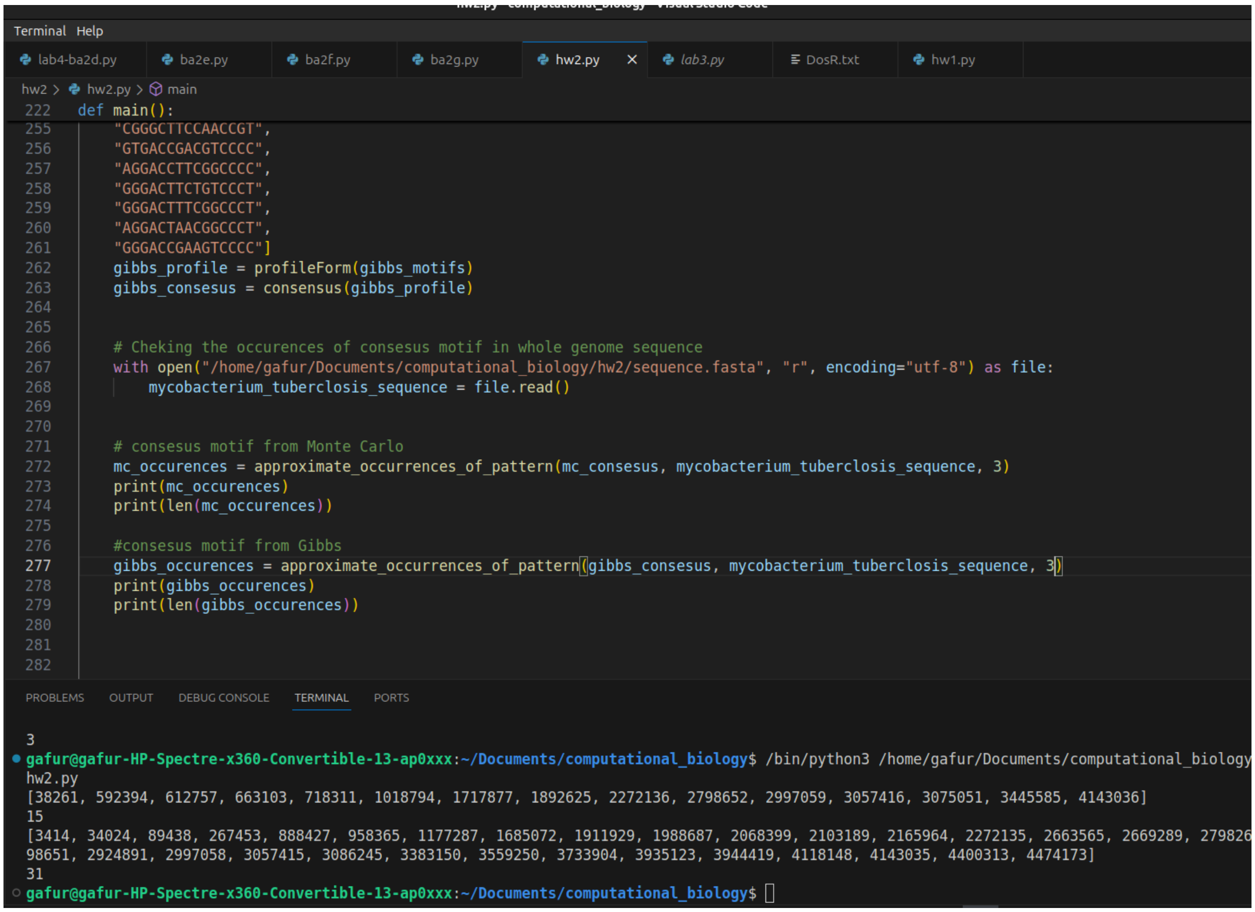


```python
def main():
    mc_motifs = [
    "GGACTTCAGGCCCTA",
    "GGTCAAACGACCCTA",
    "GGACGTAAGTCCCTA",
    "GGGCTTCCAACCGTG",
    "GGCCGAACGACCCTA",
    "GGACCTTCGGCCCCA",
    "GGACTTCTGTCCCTA",
    "GGACTTTCGGCCCTG",
    "GGACTAACGGCCCTC",
    "GGACGTCCGCGACGA"]
    mc_profile = profileForm(mc_motifs)
    mc_consesus = consensus(mc_profile)

    gibbs_motifs = [
    "GGGACTTCAGGCCCT",
    "GGGTCAAACGACCCT",
    "GGGACGTAAGTCCCT",
    "CGGGCTTCCAACCGT",
    "GTGACCGACGTCCCC",
    "AGGACCTTCGGCCCC",
    "GGGACTTCTGTCCCT",
    "GGGACTTTCGGCCCT",
    "AGGACTAACGGCCCT",
    "GGGACCGAAGTCCCC"]
    gibbs_profile = profileForm(gibbs_motifs)
    gibbs_consesus = consensus(gibbs_profile)

    print("consesus of the motifs from MC Algo: ", mc_consesus)
    print("consesus of the motifs from Gibbs Algo: ", gibbs_consesus)
```

```
gafur@gafur-HP-Spectre-x360-Convertible-13-ap0xxx:~/Documents/computational_biology$ /bin/python3 /home/gafur/Do
hw2.py
consesus of the motifs from MC Algo:  GGACTTACGGCCCTA
consesus of the motifs from Gibbs Algo:  GGGACTTACGGCCCT
gafur@gafur-HP-Spectre-x360-Convertible-13-ap0xxx:~/Documents/computational_biology$
```

Mismatch Analysis

To further validate the identified motifs, I searched for occurrences of the consensus sequences with up to three mismatches. This step helps identify slight differences in the motif that may be caused by natural genetic variations, small sequencing errors, or minor changes over time, while still keeping the important part of the motif intact. Identifying these near-matches helps in understanding motif conservation across different genes and strengthens confidence in the discovered patterns.



```python
def main():
    "CGGGCTTCCAACCGT",
    "GTGACCGACGTCCCC",
    "AGGACCTTCGGCCCC",
    "GGGACTTCTGTCCCT",
    "GGGACTTTCGGCCCT",
    "AGGACTAACGGCCCT",
    "GGGACCGAAGTCCCC"]
    gibbs_profile = profileForm(gibbs_motifs)
    gibbs_consesus = consensus(gibbs_profile)


    # Cheking the occurences of consesus motif in whole genome sequence
    with open("/home/gafur/Documents/computational_biology/hw2/sequence.fasta", "r", encoding="utf-8") as file:
        mycobacterium_tuberclosis_sequence = file.read()


    # consesus motif from Monte Carlo
    mc_occurences = approximate_occurrences_of_pattern(mc_consesus, mycobacterium_tuberclosis_sequence, 3)
    print(mc_occurences)
    print(len(mc_occurences))

    #consesus motif from Gibbs
    gibbs_occurences = approximate_occurrences_of_pattern(gibbs_consesus, mycobacterium_tuberclosis_sequence, 3)
    print(gibbs_occurences)
    print(len(gibbs_occurences))
```

```
3
gafur@gafur-HP-Spectre-x360-Convertible-13-ap0xxx:~/Documents/computational_biology$ /bin/python3 /home/gafur/Documents/computational_biology
hw2.py
[38261, 592394, 612757, 663103, 718311, 1018794, 1717877, 1892625, 2272136, 2798652, 2997059, 3057416, 3075051, 3445585, 4143036]
15
[3414, 34024, 89438, 267453, 888427, 958365, 1177287, 1685072, 1911929, 1988687, 2068399, 2103189, 2165964, 2272135, 2663565, 2669289, 279826
98651, 2924891, 2997058, 3057415, 3086245, 3383150, 3559250, 3733904, 3935123, 3944419, 4118148, 4143035, 4400313, 4474173]
31
gafur@gafur-HP-Spectre-x360-Convertible-13-ap0xxx:~/Documents/computational_biology$ 
```

Conclusion

The motif discovery process demonstrates the effectiveness of Gibbs sampling and MC over brute force methods in identifying conserved sequence elements. The consensus motifs obtained from different algorithms show slight variations but retain a common core sequence. The next steps involve evaluating the functional significance of these motifs, potentially linking them to transcription factor binding sites or regulatory mechanisms in gene expression.