Algorithms in Computational Biology – Homework Exercise 4

Publication date: *Thursday, December 24*
Due date: *Sunday, January 10, 2021 (9pm IST)*

Consider the gene structure model described in HW exercise #3. As you have discovered, this model can be described using a six-state HMM with the following transition and emission probability matrices:

	S1	S2	S3	S4	S5	S6
S1	T_{II}	T_{IG}	0	0	0	0
S2	0	0	1	0	0	0
S3	0	0	0	1	0	0
S4	0	0	0	0	1	0
S5	0	0	T_{GG}	0	0	T_{GI}
S6	T_{II}	T_{IG}	0	0	0	0

	A	T	C	G
S1	E_{IA}	E_{IT}	E_{IC}	E_{IG}
S2	1	0	0	0
S3	E_{GA}	E_{GT}	E_{GC}	E_{GG}
S4	E_{GA}	E_{GT}	E_{GC}	E_{GG}
S5	E_{GA}	E_{GT}	E_{GC}	E_{GG}
S6	0	1	0	0

Some of the transition and emission probabilities of the model are fixed to 0 or 1, and others are parameterized using the following twelve parameters:

- T_{IG} - probability of starting a gene
- T_{II} - probability of remaining in an intergenic region
- T_{GI} - probability of ending a gene
- T_{GG} - probability of remaining in a gene (transitioning to the next codon)
- E_{Ix} - probability of emitting base x in an intergenic region (x∈{A,C,G,T})
- E_{Gx} - probability of emitting base x in gene (x∈{A,C,G,T})

Note that there are only eight free parameters in the model (T_{IG}, T_{GI}, E_{IA}, E_{IT}, E_{IC}, E_{GA}, E_{GT}, and E_{GC}) because of the following restrictions:

- $T_{IG} + T_{II} = 1$
- $T_{GI} + T_{GG} = 1$
- $E_{IA} + E_{IT} + E_{IC} + E_{IG} = 1$
- $E_{GA} + E_{GT} + E_{GC} + E_{GG} = 1$

In this exercise you will implement, execute, and experiment with various algorithms for inferring parameters in this model.

a. Assume that you are given fully annotated data – a DNA sequence and state annotations for every base (X, S). **Write down the log-likelihood** as a function of the twelve model parameters specified above and the number of different transitions and emissions in the observed path S. Simplify this function so that it becomes a function of a small number of sufficient statistics.

Follow these guidelines:

1) Express the log likelihood, $\log(P(S, X|\Theta))$, as a function of the twelve model parameters and the counts of the 18 possible emissions and 9 possible transitions (those with probability > 0).
2) Some of these counts are associated with probability 1, so they end up being omitted. The remaining terms can be expressed together as follows:

$$\log(P(S, X|\Theta)) = f_0(S, X) + \sum_p f_p(S, X) \log(p)$$

where p is one of the twelve model parameters, $f_p(S, X)$ is the **sufficient statistic** associated with p, and $f_0(S, X)$ is a term that is not associated with any parameter (it could be 0). As a result, you should get twelve sufficient statistics, which should be simple functions of transitions and emission counts.

b. Derive **maximum likelihood estimates (MLEs)** for the model parameters in the scenario of completely annotated sequences (X, S). Use the expression you derived in (a) for the log likelihood and group parameters into four groups based on the restrictions specified in the bottom of the previous page.

c. Implement the two parameter inference algorithms we saw in class: **Viterbi training** and **Baum-Welch**. Write a single program for both algorithms. The program should receive the following arguments:

- an observed sequence *X*
- an indicator for the inference method – **V** Viterbi training; **B** Baum-Welch
- initial values for the free model parameters $T_{\mathrm{IG}}^{(0)}, T_{\mathrm{GI}}^{(0)}, E_{\mathrm{IA}}^{(0)}, E_{\mathrm{IT}}^{(0)}, E_{\mathrm{IC}}^{(0)}, E_{\mathrm{GA}}^{(0)}, E_{\mathrm{GT}}^{(0)}, E_{\mathrm{GC}}^{(0)}$ (in this order).

Your program should run the specified inference algorithm on the input sequence *X* starting from the initial parameter values. The halting condition should be when the traced score is improved by **less than ε=0.00001**. The traced score is the log-likelihood for Baum-Welch and Viterbi score for Viterbi training. Use natural log – $\ln(\,)$.

The program should produce a trace of the eight free model parameters and the associated score. See execution examples in the next page.

Additional important implementation notes:

- You may assume that the initial state of the HMM is S1 with probability 100%. The HMM path may end with any state.
- Write your code clearly and provide detailed documentation for the various functions / code blocks. Submit the code as appendix to this assignment.
- In both algorithms make sure to use the formulas you received in (b) above for the parameter update step.
- Make sure to hold log-probabilities in your dynamic programming matrices, and when necessary apply the "log-of-sum-of-exponents" technique we saw in class (lecture #6). Notice that when computing expected sufficient statistics for the Baum-Welch algorithm (transition and emission counts), you need to convert values from log-probabilities to actual probabilities. Make sure that you are exponentiating probabilities that are <u>conditional</u> on the data and <u>not joint probabilities</u> with the data (e.g. $P(S|X,\Theta)$ and not $P(S,X|\Theta)$). This is because joint probabilities with the data are very small and could evaluate to 0, and the expected counts you need for the Baum-Welch algorithm require conditional probabilities, which are larger.
- When debugging the procedures for computing the forward / backward matrices, it is useful to ensure that the likelihood can be computed at every column of these matrices (as you did in HW #3).
- Another useful validation for the Baum-Welch algorithm is to make sure that the sum of expected transition and emission counts per position is 1.

Execution examples:

```
==> ./gene-hmm-train AAATTTTATTACGTTTAGTAGAAGAGAAAGGTAAACATGATGG V \
                     0.4 0.4 0.1 0.2 0.3 0.4 0.3 0.2
A A A T T T T A T T A C G T T T A G T A G A A G A G A A A G G T A A A C A T G A T G G
-------------------------------------------------------------------------------------
|  T_IG  T_GI  E_IA  E_IT  E_IC  E_GA  E_GT  E_GC    score (Viterbi score)
|  0.40  0.40  0.10  0.20  0.30  0.40  0.30  0.20     -64.9658  |
|  0.29  0.18  0.50  0.00  0.00  0.39  0.33  0.06     -53.1671  |
|  0.15  0.22  0.58  0.00  0.00  0.33  0.41  0.07     -51.9011  |
|  0.15  0.22  0.58  0.00  0.00  0.33  0.41  0.07     -51.9011  |


Done.
```

```
==> ./gene-hmm-train AAATTTTATTACGTTTAGTAGAAGAGAAAGGTAAACATGATGG B \
                     0.4 0.4 0.1 0.2 0.3 0.4 0.3 0.2
A A A T T T T A T T A C G T T T A G T A G A A G A G A A A G G T A A A C A T G A T G G
-------------------------------------------------------------------------------------
|  T_IG  T_GI  E_IA  E_IT  E_IC  E_GA  E_GT  E_GC    score (log likelihood)
|  0.40  0.40  0.10  0.20  0.30  0.40  0.30  0.20     -61.9375  |
|  0.30  0.28  0.42  0.24  0.00  0.39  0.29  0.07     -51.7691  |
|  0.24  0.30  0.52  0.17  0.00  0.35  0.33  0.08     -50.7574  |
|  0.19  0.32  0.56  0.11  0.00  0.31  0.38  0.08     -49.9532  |
|  0.16  0.30  0.58  0.09  0.00  0.28  0.42  0.09     -49.6298  |
|  0.14  0.28  0.58  0.08  0.00  0.27  0.44  0.09     -49.5454  |
|  0.13  0.27  0.58  0.08  0.00  0.27  0.45  0.09     -49.5224  |
|  0.13  0.26  0.58  0.08  0.00  0.27  0.45  0.09     -49.5152  |
|  0.13  0.26  0.58  0.08  0.00  0.27  0.45  0.09     -49.5119  |
|  0.12  0.26  0.58  0.08  0.00  0.26  0.46  0.09     -49.5091  |
|  0.12  0.26  0.57  0.08  0.00  0.26  0.46  0.09     -49.5058  |
| .
| .          [ RUNS FOR ~50 ITERATIONS ]
| .
|  0.07  0.29  0.54  0.11  0.03  0.21  0.56  0.08     -49.3064  |
|  0.07  0.29  0.54  0.11  0.03  0.21  0.56  0.08     -49.3064  |
|  0.07  0.29  0.54  0.11  0.03  0.21  0.56  0.08     -49.3064  |
|  0.07  0.29  0.54  0.11  0.03  0.21  0.56  0.08     -49.3064  |
|  0.07  0.29  0.54  0.11  0.03  0.21  0.56  0.08     -49.3064  |


Done.
```

After implementing the two algorithms, we wish to execute each algorithm on the three following DNA sequences extracted from the virus:

*X*1:

```
AAATTTTATTACGTTTAGTAGAAGAGAAAGGTAAACATGATGGTTCAGTGGTGCTAGATGAACAAACAATT
ATAAAATAAAATGAAGTATTTGTATAGAA
```

*X*2:

```
CCCCCCAGGGGGGGGGGGGGTCCCCCCCCCCCCCCCCCCCCCCCAGGGGGGGGGGGGGGGGGGTCCCAGGGGG
GGGGGGGGGGTCCAGGGTCCCCCCCCCCC
```

*X*3:

```
CGCACACGTCCTTGAGGGCAGTTTTTTTGTCGCCCCCACGATTTTTCTCGGCCGCAGTTCCCGTTTTTTTT
TGTTTTTTTTGTTGGCCTCTGGTTTTCTACGAGGCCGGGGAGAGGCCGGGGCGGCAGATTTTCTTGTTTTT
CAGGATTGCTGGTTTGCTCAGTGTTTTTCTTCTTTGTTTGGCTGTGCCGGAAGAGATG
```

d. Use your program to find for each of the three DNA sequences specified above the values of the parameters that maximize the Viterbi score (use Viterbi training). Run your program from multiple starting points for each input instance to increase your confidence. In your solution, write the best parameter set you found for each sequence, and describe your strategy – which starting points you chose, and how you chose them. Your strategy may use random sampling. Your solution should include a brief report of your experiments (~ one page) and a separate file named `traces-viterbi.txt` containing a well-marked list of the traces for the runs you made to support your final conclusion.

e. Use your program to find for each of the three DNA sequences specified above the values of the parameters that maximize the log likelihood (use Baum-Welch). Run your program from multiple starting points for each input instance to increase your confidence. In your solution, write the best parameter set you found for each sequence, and describe your strategy – which starting points you chose, and how you chose them. Your strategy may use random sampling. Your solution should include a brief report of your experiments (~ one page) and a separate file named `traces-baum-welch.txt` containing a well-marked list of the traces for the runs you made to support your final conclusion.

f. In (d) and (e) above you inferred model parameters for each of the three DNA sequences using two different optimization criteria. Compare the two sets of parameter values that you inferred for each of the three data sets. Compute the Viterbi score and log-likelihood for each inferred model. How similar are the two models you inferred for each inputs sequence? Clearly describe your findings and if you see differences then briefly explain what leads to these differences.

Submission Instructions:

- Submit your work on the course **Moodle website** by Sunday, Jan 10, 2021 @21:00.
- Submit written answers for items (a), (b), (d), (e), and (f).
- Submit your documented code for (b) in a separate file (or files).
- Submit your traces for (e) and (f) in separate files, as specified. For each trace specify the command line used to obtain it and clearly mark the final result (parameter values and score).
- Type your solutions or write legibly and scan. **If you scan, make sure the scan came out fine.**
- **Submit your work in pairs!** One student should submit a solution file with both of your student IDs specified. The other student should submit a simple text file with your two student ids, to help us match back the grade to both of you.
- If you consult with other pairs on ideas, specify their names clearly on the first page and make sure that they **acknowledge your collaboration** as well.
- You have two weeks to complete the assignment. **Plan your time wisely**. Extensions due to special circumstances, will be granted only upon **request by e-mail at least 48 hours** before the deadline. No last minute extensions!
- Please post any questions that you have on the course Piazza website: https://piazza.com/idc.ac.il/fall2020/cs3571/.