COMP 5970/6970-004
Computational Biology: Genomics and Transcriptomics
Lecture notes 6: 2/1/2022

Haynes Heaton

Spring, 2022

Lecture Objectives

- Estimating statistical model parameters
  - Maximum Likelihood methods
    - Expectation Maximization (EM)
    - Gradient descent
  - Posterior distribution of parameters
    - Variational Inference
    - Markov Chain Monte Carlo (MCMC)
- Homework 1 extra credit solution
- Semi-Global Alignment
- Local Alignment

Estimating statistical model parameters

When the generation of data is not trivial, we still may be able to model it with some compound distribution or mixture of distributions. If we do not know the parameters of these distributions, they may be estimated via statistical inference.

0.1 Maximum Likelihood Estimation (MLE

Last class, we did just this on a binomial mixture model with expectation maximization by starting with random model parameters and iteratively finding the posterior probability of each data point to each distribution and then updating the parameters of those distributions to improve

the total likelihood. This method only guarantees finding a local optima of likelihood given a set of parameters. In certain models, this will always be the global optimum, but not all. One way to get around this is to restart the algorithm with different random parameter initializations and choosing the parameter set in which the likelihood was maximal. If the likelihood function is differentiable, another method available is gradient descent, which may be used to minimize the negative log likelihood thus maximizing the likelihood. This is also only guarantees finding a local optimum given a random initialization. In addition to this lack of global guarantee, we only find the single most likely set of parameters and do not model our uncertainty in those parameters.

0.2 Posterior distribution of parameters

There are a number of methods to try to model the uncertainty in the parameters as well as attempting to overcome local maxima in our likelihood landscape. Variational Inference is a method that minimizes the Kullback Leibler divergence between the data and the model distribution giving a distribution on each model parameter. However, this method also has limitations in that it also does not find a global maximum in the likelihood (or the full posterior distribution of parameters) and can only give smooth distributions over model parameters. Markov Chain Monte Carlo (MCMC) uses a probabilistic stochastic search process that, with infinite iterations, will converge to the posterior distribution. MCMC has the downside of being slow and only guaranteeing convergence to the posterior distribution with infinite iterations. I just want you all to know that these methods exist rather than knowing how they work which is out of the scope of this class. Different programming packages implement these algorithms. STAN is a domain specific language for statistical models which implements gradient descent, variational inference, and MCMC. Tensorflow probability implements gradient descent and variational inference and is a standard tool in many machine learning workflows.

Homework 1 extra credit solution

As a reminder, the problem was
7. Describe a dynamic programming approach to finding the largest checkerboard pattern in a matrix of zeros and ones where zero represents white and one represents black (checkerboard pattern is just alternating 0s and 1s, it does not matter which color starts or ends the pattern). Checkerboard must be square. Don't spend over an hour on this. If you find it difficult, come to my office hours after class and we can work through it together. Starting with 7a may help. And coming up with rules and testing them out may lead to the correct rules so use the example matrix below.

7a. What is the base case for this algorithm?

Here is an example matrix to help develop intuition and test your method.

0	1	1	1	0	1
1	0	1	0	1	0
0	1	0	1	0	1
1	0	1	0	1	0
1	0	0	1	1	0

Generally in dynamic programming you will have a table that keeps track of the optimal solution(s) up to this point. We need to decide what our base case solution is. In this case, the simplest base cases to use would be a $0x0$ size checkerboard or a $1x1$ checkerboard. For this example, I will use $1x1$. So a single cell of 1 or 0 are both $1x1$ checkerboards. Now we need to decide what the values in the DP table represent. Similar to our global alignment problem, we could have a DP table of the same size as the input matrix. In the global alignment problem, the value was the score of the alignment up until now. Here it is natural to make the DP table keep track of the largest checkerboard cell i, j is the bottom right corner of. If that is the case, the left column and top row are all the base case of a $1x1$ checkerboard. We will denote the input matrix I and the DP table S for score. So the DP table S is initialized as

1	1	1	1	1	1
1					
1					
1					
1					

Now at each $i = 1..n$ and $j = 1..m$, we can check the cell above and to the left of it in the input matrix and make sure they are different and the one diagonal to the upper left and make sure they are they same. If that is the case, this square is the bottom right of a checkerboard that is $min(S[i-1][j-1], S[i][j-1], S[i-1][j]) + 1$. If this isn't the case, this is only the base case

checkerboard of $1x1$. So our formal recurrence relation is

$$S[i,j] = \begin{cases} min(S[i-1,j-1], S[i-1][j], S[i][j-1]) + 1 \\ \quad \text{if } I[i][j] = I[i-1][j-1] \\ \quad\quad \text{and } I[i][j] \neq I[i-1][j] \\ \quad\quad \text{and } I[i][j] \neq I[i][j-1] \text{ (continuation of a checkerboard)} \\ 1 \text{ otherwise} \end{cases}$$

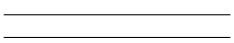
So filling the DP table out, we have

1	1	1	1	1	1
1	2	1	1	2	2
1	2	2	2	2	3
1	2	3	3	3	3
1	1	1	2	1	1

So we have several ties for different locations of a $3x3$ checkerboard in the given input matrix.

1 Semi-Global Alignment

Recall our global alignment algorithm from lecture 1. One downside of this algorithm is that it assumes there is full overlap between two sequences such as shown below.



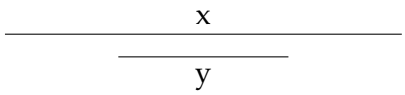
But it is very common to want to compare two sequences which do not fully overlap. Perhaps one sequence is fully encompassed by another sequence. Or perhaps the two sequences overlap in their sequence but have overhanging ends.

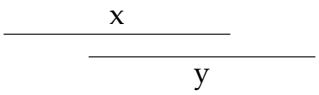
This is known formally as semi-global alignment and informally as overlap alignment. This requires just two differences from our global alignment algorithm.

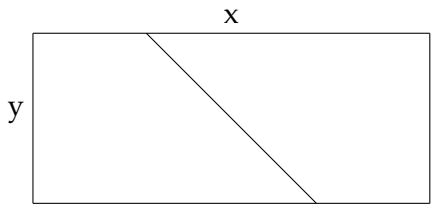
1. Where the Viterbi backtrace can begin and end changes from bottom right to top left to the maximum value on the bottom row or right column and goes until it hits the left column or top row.

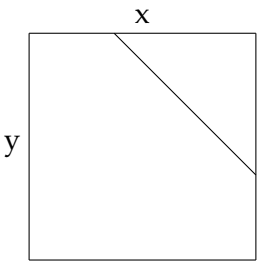
2. The initialization of the top row and left column become all 0s as instead of them indicating indels, they indicate the start of an alignment.

Depending on how our sequences overlap, we will have a few different patterns of where the Viterbi backtrace begins and ends.

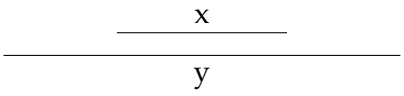


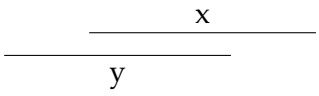


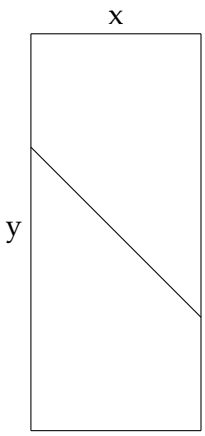


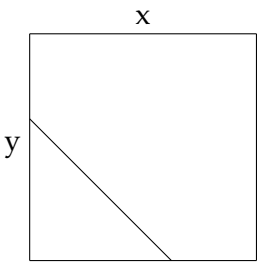


and









So in this case, we initialize the top row and left column with 0s indicating possible starts of an overlap alignment.

		C	A	G	C	T	G	G
	0	0	0	0	0	0	0	0
T	0							
A	0							
C	0							
A	0							
G	0							
G	0							
A	0							
T	0							
T	0							

And we proceed as we did in the global alignment for filling out the score matrix. As a reminder, the recurrence relation is

$$m[i,j] = max \begin{cases} m[i-1,j-1]+1 & \text{if} \quad s_1[i] = s_2[j] \text{ (match)} \\ m[i-1,j-1]-1 & \text{if} \quad s_1[i] \neq s_2[j] \text{ (mismatch)} \\ m[i-1,j]-1 \text{ (indel)} & \\ m[i,j-1]-1 \text{ (indel)} & \end{cases}$$

		C	A	G	C	T	G	G
	0	0	0	0	0	0	0	0
T	0	-1	-1	-1	-1	1	-1	-1
A	0	-1	0	-1	-2	-2	-1	-2
C	0	1	0	-1	0	-1	-2	-2
A	0	0	2	1	0	-1	-2	-3
G	0	-1	1	3	2	1	0	-1
G	0	-1	0	2	2	1	2	1
C	0	1	0	1	3	2	1	1
T	0	0	0	0	2	4	3	2

Now we take the maximum value in the bottom row or right column to begin the backtrace algorithm. This is the value 4. So we have the following backtrace.

		C	A	G	C	T	G	G
	0	0	0	0	0	0	0	0
T	0	-1	-1	-1	-1	1	-1	-1
A	0	-1	0	-1	-2	-2	-1	-2
C	0	1	0	-1	0	-1	-2	-2
A	0	0	2	1	0	-1	-2	-3
G	0	-1	1	3	2	1	0	-1
G	0	-1	0	2	2	1	2	1
C	0	1	0	1	3	2	1	1
T	0	0	0	0	2	4	3	2

Creating an alignment of

```
 CA-GCTGG
TCAGGCT
```

Local Alignment - AKA Smith-Waterman

The next type of alignment is when you expect two sequences to only align in a subsequence of both sequences rather than an overlap or a full length alignment. Again, this is a small difference from the previous alignment algorithms.

1. Whenever an alignment score would go negative, instead it starts a new alignment with score 0
2. The Viterbi backtrace starts and the maximum score in the entire matrix and goes back to the first 0 the backtrace encounters

$$m[i,j] = max \begin{cases} m[i-1,j-1]+1 & \text{if } \; s_1[i] = s_2[j] \text{ (match)} \\ m[i-1,j-1]-1 & \text{if } \; s_1[i] \neq s_2[j] \text{ (mismatch)} \\ m[i-1,j]-1 \text{ (indel)} & \\ m[i,j-1]-1 \text{ (indel)} & \\ 0 \text{ (start new alignment)} & \end{cases}$$

So we once again initialize the top row and left column with alignments starting with a score of 0.

		G	G	C	G	A	A	C	A
	0	0	0	0	0	0	0	0	0
T	0								
A	0								
G	0								
C	0								
G	0								
A	0								
T	0								
T	0								

		G	G	C	G	A	A	C	A
	0	0	0	0	0	0	0	0	0
T	0	0	0	0	0	0	0	0	0
A	0	0	0	0	0	1	1	0	1
G	0	1	1	0	1	0	0	0	0
C	0	0	0	2	1	0	0	1	0
G	0	1	1	1	3	2	1	0	0
A	0	0	0	0	2	4	3	2	1
T	0	0	0	0	1	3	3	2	1
T	0	0	0	0	0	2	2	2	1

Creating an alignment of

```
GCGA
GCGA
```

with the rest of the sequence unaligned.