Series 4

Genomics and bioinformatics - Week 4 - October 16, 2012

1 Hidden Markov Model

The goal of this exercise is to design a Hidden Markov model. *Plasmodium Falciparum* (protozoan parasite that causes malaria in humans) has a GC content of about 20%, and a genome length of 23Mb. Suppose that we are interested in a protein that has a strong affinity with GC-rich isochores (long regions of DNA with a relatively homogeneous GC content, which tend to be more flexible and contain more genes), and thus it is interesting to find GC-rich isochores to discover potential binding sites of the protein. In the case of *P. falciparum*, we know that there is only one large (7 Kb) GC-rich isochore with a 50% GC content - but suppose we do not know where it is.

1. Draw a Hidden Markov Model that reflects the situation: specify hidden states and observed variables.

2. What are the emission probabilities from each state?

3. Taking uniformly a random position in the genome, what is the probability that it belongs to the GC-rich isochore? Call this probability $x = P(I)$, the complementary $1-x = P(N)$.

4. Write p and q for the transition probabilities between the two states N and I. Write $P(I|N)$, $P(N|I)$, $P(I|I)$, $P(N|N)$, $P(I)$ and $P(N)$ as functions of x, p and q, and try to solve for p and q.

5. Once in a given state, consider the random variable X that describes the number of steps until one leaves the state. What is its distribution, and its mean?

6. Estimate p and q.

2 Reading frame

In this exercise you are given a nucleotide sequence (`"sequence_ex2.fasta"`) which contains a coding region somewhere[1]. You have to deduce the correct (longest) reading frame of this coding region.

The general procedure to find the right frame for reading a nucleotide sequence is to convert the nucleotide sequence into the corresponding possible amino acid sequences and see which one makes the most sense. As you know, the base pairs are read three by three and translated into amino acids. This can be done on the forward strand or the complementary strand. One can hence read a sequence in six different ways, depending on where one starts ("phase").

So, to convert a base pair sequence (e.g. `CAGATTCTC`...) to a amino acid sequence (e.g. `GWLPHLQRI`...) you cut the base pair sequence in pieces of 3 nucleotides (e.g. `'CAG', 'ATT',...`),

[1] From the yeast TCP1-beta gene. The complete sequence can be found at `http://www.ncbi.nlm.nih.gov/sites/entrez?cmd=Retrieve&db=nucleotide&dopt=GenBank&list_uids=1293613`

and use a conversion table that links any possible 3-mer to one of the 21 amnio acids. For instance, `CAG` codes for glutamine.

To do this, you can use you favorite programming language - choosing Python or R will make you assistants more willing to help you, and we advise Python for this kind of work.

2.1 All 3-mers

To build the conversion table that links 3-mers to amino acids, we first need to build an exhaustive list of 3-mers. Write the code that takes as input the list of the four nucleotides and generates as output all the possible permutations of size 3 (64 elements):

```
bases = ['t', 'c', 'a', 'g']
codons = ['ttt', 'ttc', 'tta', 'ttg', 'tct', 'tcc', 'tca', 'tcg', 'tat', ... ]
```

2.2 3-mer to amino acid

We can now map each 3-mer to its amino acid. If you build the list in the same order as above, the corresponding amino acids are the following:

```
aminos = "FFLLSSSSYY**CC*WLLLLPPPPHHQQRRRRIIIMTTTTNNKKSSRRVVVVAAAADDEEGGGG"
codon_to_aminoacid = {'aaa': 'K', 'aac': 'N', 'aag': 'K', 'aat': 'N', 'aca': 'T', ... }
```

2.3 Sequence to protein

You can now write the function that takes a nucleotide sequence as entry and outputs a protein sequence:

```
def translate(seq):
   <code>
```

You should be able to use it like this:

```
translate('cagattctc')
>>> QIL
```

Construct also a second function `complementary` that takes a nucleotides sequence and returns its reverse complement.

2.4 Find the correct reading frame

You can now load the file `"sequence_001.txt"` and call the functions you wrote in the last step with the six different possible frames and find the longest reading frame.

2.5 Some useful features for this exercise (Python)

- Dictionaries (`dict`) are *the* mapping type in Python.
  One can make a dictionary from lists using `dict(zip(`*list*1, *list*2`))`.
- Strings: `replace, split`.
  Lists: `''.join(`*list*`)` transforms a list of characters into a string (`list(`*string*`)` does the contrary); *list*`.append()` adds an element at the end of the list.
- *Biopython* probably provides tools to do all this rapidly (unchecked).

3 BLAST

A common use of the BLAST tool is to identify the function of an unknown sequence. You have been provided with the sequence of a DNA fragment, `"fragment_007.fasta"` from an unknown micro-organism. Your aim is to use the NCBI BLAST programs to determine what kind of protein is `fragment_007` is likely to encode:

`http://blast.ncbi.nlm.nih.gov/`

3.1 Nucleotide BLAST - blastn

1. Perform a nucleotide BLAST using `fragment_007.fasta` using the default parameters. Do you get any matches to `fragment_007`? Give two possible reasons why this does not work.

2. Now choose a larger database (*Nucleotide collection (nr/nt)*) and allow for *More dissimilar sequences*. You can also change the scores for match/mismatch and gap penalties in the *Algorithm parameters*. Run BLAST again and record the top hit.

3. You can sort the BLAST results based on a number of alignment statistics. What criteria, i.e. which statistics would you consider in order to assess the statistical significance of BLAST hits? Justify your choice.

4. Do you have any significant hits suggesting a possible function for `fragment_007`?

3.2 Protein BLAST - blastp

Using the python function from the previous exercise, obtain the amino acid sequences for `fragment_007.fasta` in three reading frames. Choose the appropriate reading frame and save the corresponding amino acid sequence in a text file, say `aa_007.fasta`.

1. Perform a protein BLAST with `aa_007.fasta` using the default parameters. Are any well-known protein domains found? Record the top hit.

2. What is the possible function of the protein encoded by `fragment_007`?

3. Which species is most predominant in your BLAST output?

4. Is there a BLAST program that would have given you the same results as BLASTp using the nucleotide sequence of `fragment_007` as input?

5. How do results from BLASTp compare with the results from BLASTn? How do you explain the differences?

3.3 Finding orthologs

You have been provided with the sequence of the Pho2p protein from the famous yeast, *Saccharomyces cerevisiae*: `pho2p_cerevisiae.fasta`. Run BLAST to find out whether any putative orthologs of Pho2p are present in another, not so famous yeast, *Candida glabrata*.

In a 2009 publication[2] comparing the phosphate signal transduction in *S. cerevisiae* and *C. gabrata*, authors identified `CAGL0L07436g` as the ortholog of Pho2p in *C. gabrata*. Are the findings of the paper consistent with your observations?

3.4 Some BLAST tips

- This web page provides useful guidelines for BLAST usage and explains the different flavors of BLAST: `http://www.clcbio.com/index.php?id=995`
- *Reciprocal Best Hit*

  This is a simple and commonly used test for predicting orthologous sequences.

  Genes A (from species X) and B (from species Y) will be considered as (putative) orthologs if (1) a search of similar sequences of A in species Y yields as the best hit B, and (2) a search of similar sequences of B in species X yields as the best hit A.
- One can automate large-scale, numerous BLAST requests using Biopython: `http://biopython.org/DIST/docs/tutorial/Tutorial.html`, Chapter 7.

[2]Kerwin and Wykoff, 2009: `http://www.genetics.org/content/182/2/471.full`