

Series 5

Genomics and bioinformatics - Week 6 - October 18, 2011

1 Hidden Markov Model

The goal of this exercise is to design and analyze 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 now that our protein of interest 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). We are interested in finding isochores to discover potential binding sites of the protein. In the case of *P. falciparum*, we know that only one 7Kb DNA isochore has a 50% GC content - but suppose we don't know where it is.

1. Draw a Hidden Markov Model that reflects the situation: identify hidden states and observed variables.
2. What are the emission probabilities from each state?
3. What is the probability, taking a random position in the genome, to be in the isochore? Call this probability $x = P(I)$, the complementary $1 - x = P(N)$.
4. Call p, q the transition probabilities between 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 the relations between them (Bayes' formula).
5. Once in a given state, consider the random variable that describes the number of (Bernoulli) 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_001.txt`") which contains a coding region somewhere. You have to deduce what is the 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`', ...), and use a conversion table that links any possible 3-mer to one of the 21 amino acids. For instance, `CAG` codes for glutamine.

To do this, you can use your favorite programming language - choosing Python or R will make your 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.

The output should start like this and have 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 built the list in the same order as above, the corresponding amino acids are the following:

```
amino = "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): .....
```

You should be able to use it like this:

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

2.4 Find the correct reading frame

You can now load the file "sequence_001.txt" and call the function you wrote in the last step with the six different possible frames and decide which one is the right one.

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(list1, list2))`.
- Strings: `replace`, `lower`, `upper`, `strip`, `split`.
`"".join(list)` transforms a list of characters into a string;
`list(string)` does the contrary.
- 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

Perform a nucleotide BLAST using `fragment_007.fasta` .

1. Do you get any matches to `fragment_007`? Which parameters did you use? Record the alignment statistics for the top hits.
2. Extract the sequence of the hit with the highest query coverage (this may not necessarily be the top hit) and perform another nucleotide BLAST, using the same parameters. Record the alignment statistics for the top hits.
3. What changes do you observe in the E-values? To which parameter would you attribute the these changes?
4. What is the default threshold for the E-value on NCBI BLAST?
5. 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 `aa_007.fasta` format.

Perform a protein BLAST with `aa_007.fasta`

1. Are any well-known protein domains found?
2. Do you get any significant hits? Record the alignment statistics for the top hits.
3. What is the possible function of the protein encoded by `fragment_007`?
4. Which species is most predominant in your BLAST output?
5. Is there a BLAST program that would have given you the same results as BLASTp using the nucleotide sequence of `fragment_007` as input?
6. How do results from BLASTp compare with the results from BLASTn?

3.3 Finding orthologs

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

In a paper comparing the phosphate signal transduction in *S. cerevisiae* and *C. glabrata* (Genetics 182:471-9,2009), the authors identified the ortholog of Pho2p in *C. glabrata*.

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 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.