Series 5

Genomics and bioinformatics - Week 6 - October 23, 2012

This series is about phylogenetic trees and a complement to HMMs.

1 Phylogenetic trees

Circadian rhythms (often referred to as the "body clock") are ubiquitous in most life forms. The "period" family of genes is an essential component of this clock.

The table below provides BLAST scores of pairwise alignments for six period family proteins from three different species,

1. **dm_per** from *Drosophila melanogaster*, i.e. fruit fly,
2. **dr_per2** and **dr_per3** from *Danio rerio*, i.e. zebra fish,
3. **mm_per1**, **mm_per2** and **mm_per3** from *Mus musculus*, i.e. mouse.

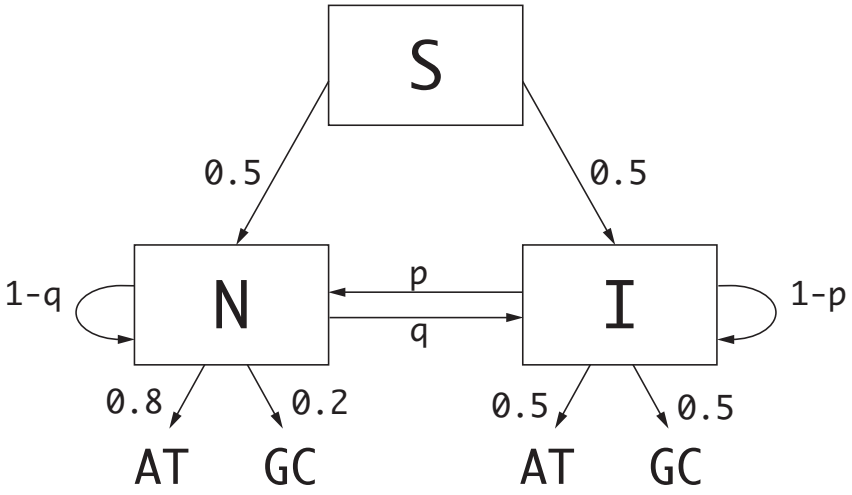
	dm_per	dr_per2	dr_per3	mm_per1	mm_per2	mm_per3
dm_per	-	116	125	129	123	121
dr_per2	114	-	661	887	947	497
dr_per3	123	644	-	667	666	699
mm_per1	126	745	572	-	640	80
mm_per2	126	984	680	832	-	455
mm_per3	124	499	709	537	488	-

1. Based on the table above, draw the gene tree for the six period proteins (dm_per, dr_per2, dr_per3, mm_per1, mm_per2, mm_per3).
2. Using this tree, give examples of:
   - an orthologous pair,
   - a paralogous pair in the same species, and
   - a paralogous pair in different species.

2 Another HMM

2.1 The Viterbi algorithm

In series 4, you designed a Hidden Markov Model. We shall now use this model to actually find the GC-rich isochore region. However, for computational reasons, the genome and GC-rich region lengths are set to 100 and 10, respectively, leading to $p = 1/10$ and $q = 1/90$. We have also included a starting state S. See the figure below.



The Hidden Markov Model

In the course, you have seen the Viterbi algorithm based on the following scoring function:

$$F_{n,s} = E(T_n, s) \cdot \max_{\sigma \in \{S,N,I\}} \{F_{n-1,\sigma} \cdot M(\sigma, s)\} \ , \ n = 1, 2, \ldots$$

The initial condition is $F_{0,\sigma} = 1$ if $\sigma = S$ and $F_{0,\sigma} = 0$ otherwise. Use this formula to fill in the following table by hand and then use backtracking to find the most probable hidden state sequence associated to the observed sequence "ATC":

	-	A	T	C
S				
N				
I				

The Table

2.2 Using R

In this exercise, we show you how to use an existing package to model a problem with an HMM and solve it. There exist such packages for most programming languages, but here we arbitrarily chose R.

2.2.1 Get started

Start R, then download, install and import the HMM package:

```
install.packages("HMM")
library(HMM)
```

Create an vector `states` of hidden states, an vector `emissions` of emission, a vector `initProb` with the probabilities of the initial state, a matrix `transProb` with the transition probabilities, a matrix `emissionProb` with the emission probabilities:

```
states <- c("N","I")
emissions <- c("C","G","A","T")
initProb <- c(0.5, 0.5)
transProb <- matrix(c(89/90, 0.1, 1/90, 0.9), 2)
emissionProb <- matrix(c(0.1, 0.25, 0.1, 0.25, 0.4, 0.25, 0.4, 0.25), 2)
```

Create the model with these parameters:

```
hmm <- initHMM(states, emissions, initProb, transProb, emissionProb)
hmm  # prints a summary of the model
```

Using the model created above, generate random observations of length 100:

```
simulation <- simHMM(hmm,100)
simulation  # prints the chosen hidden states and emissions sequences
```

2.2.2 Visualize

Copy and paste the following code to visualize the situation. The plot on the top shows the observed sequence, the one at the bottom shown the hidden states:

```
obsToVal <- function(obs){
if((obs == "A") | (obs=="T")){return(2)}
else {return(4)}
}
stateToVal <- function(st){
if(st == "N"){return (1)}
if(st == "I"){return (3)}
else {return ("NA")}
}
par(mfrow=c(2,1))
plot(sapply(simulation$observation, obsToVal),pch=19,ylim=c(0,8),
col=sapply(simulation$observation, obsToVal),cex=.5)
legend("topright",c("A/T","G/C"),pch=19,col=c("red","blue"))
plot( sapply(simulation$states, stateToVal),type="l",ylim=c(0,5))
```

2.2.3 Forward, Backward algorithms

Here are the commands to run the forward and backward algorithms. What is the probability of the observed sequence, given the model ?

```
logForwardProb <- forward(hmm, simulation$observation)
exp(logForwardProb)
logBackwardProb <- backward(hmm, simulation$observation)
exp(logBackwardProb)
```

2.2.4 Viterbi

The function `simHMM` returns a path of (hidden) states and associated observations. Suppose now that we have an observed sequence and that we want to compute the corresponding most probable sequence of hidden states. Try first to guess where are the "I" region(s) are in the following sequence:

```
observation <- c("A", "T", "A", "A", "A", "A", "T", "A", "A", "T", "T", "T", "T", "T",
 "A", "T", "A", "A", "C", "T", "A", "T", "A", "T", "A", "T", "G", "T", "A", "C", "A",
 "T", "A", "T", "T", "T", "T", "T", "A", "T", "T", "A", "A", "T", "T", "A", "G", "C",
 "G", "C", "T", "T", "C", "C", "G", "A", "A", "C", "T", "A", "G", "G", "A", "G", "A",
 "G", "C", "G", "T", "T", "T", "A", "T", "T", "T", "T", "T", "T", "T", "T", "A", "T",
 "T", "T", "A", "A", "A", "A", "T", "A", "T", "A", "C", "T", "T", "A", "A", "A", "A", "A")
```

Now we are going to find the most probable hidden sequence that explains these observations thanks to the Viterbi algorithm. Is it close to what you expected ?

```
viterbi <- viterbi(hmm, observation)
viterbi
```

To visualize the sequences, you may use the plot functions used in point **??**.

2.2.5 Baum-Welsch

Finally, let us admit that our transition and emission probabilities are just a first guess, but we would like to update the model parameters by training it with the Baum-Welsch algorithm:

```
bw = baumWelch(hmm, observation, 10)  # 10 iterations
print(bw$hmm)
```