Exercises 6-7
Statistics in Genetics

Alfons Edbom Devall

March 15, 2022

Contents

1 Chapter 3

1.1 Exercise 1

1.1.1 Background

Basic Local Alignment Search Tool, or BLAST is an algorithm for quickly finding regions of local similarity between sequences. BLAST allows users to quickly compare a subject protein or nucleotide sequence (often referred to as the query sequence) with huge databases of sequences to identify sequences that resemble the query sequence.

In this exercise the DNA sequence of the DNA sequence of the gene *eyeless* (X79493) from *Drosophila Melanogaster* was used in a BLASTn search. The aim of the search was to find a similar gene that exists in humans. In previous exercises both a global and local alignment has been done between *eyeless* and the human gene *aniridia* (AY707088), so it is of special interest to investigate whether *aniridia* can be detected when doing a BLASTn search with *eyeless*.

1.1.2 Methods

The DNA sequence for *eyeless* was downloaded from Genbank. After the sequence was downloaded, ncbi's BLAST-suite was used to make a BLASTn search. The database used used to search for was "Nucleotide collection (nr/nt). Since we were interested in similar sequences in humans, "Homo sapiens" was entered in the Organism field. The specific program for the BLAST-search were selected to be blastn, since we do not expect the sequences to have a very high similarity. To make the comparison to previous exercises easier, the same scoring parameters was used: 1 for a match, -1 for a mismatch as well as the gap opening cost of 3 and gap extension cost of 1 (note that these are not the default parameter settings).

1.1.3 Results and Discussion

The best alignments found in the BLAST-search was all transcript variants of the *PAX6* gene, which is another name of *aniridia*. The specific sequence (AY707088) could be found in the top 30 results. Note that only one other gene than *PAX6* was found before AY707088, which was *FLJ95740*, and they had all of the same statistics, so it might just be another splicing variant of *PAX6*. The resulting statisticts were found for AY707088: Max Score = 281, Total score = 421, Query Cover = 23%, E value = $4e-72$. Percent identity = 72.59% and

Accession Length = 1269. In figure 1, the best 50 alignments can be seen and AY707088 is one of the topmost, close to identical sequence alignments.

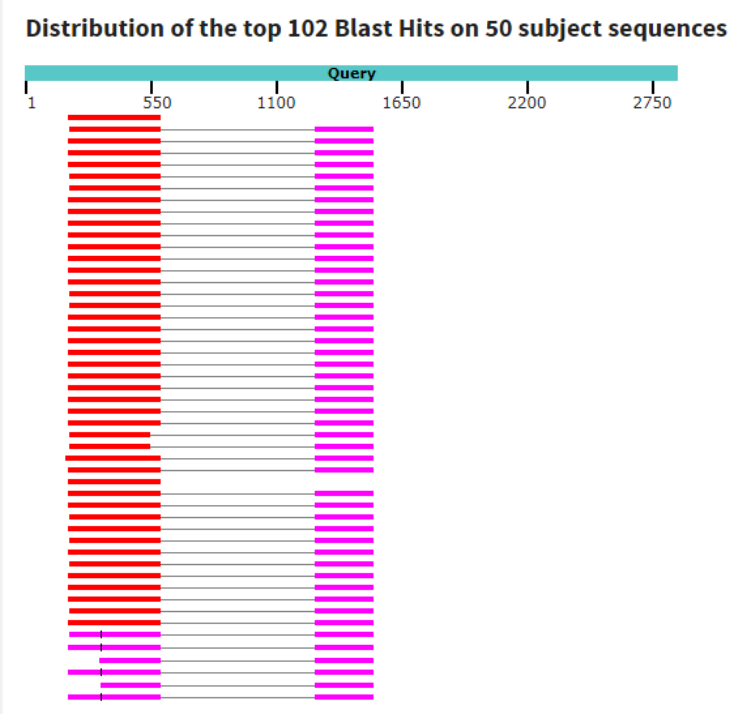

Figure 1: The 50 best alignments found by BLAST, using X79493 as the query sequence.

Since the alignment done in previous exercises was done the other way around (using *aniridia* as the template sequence), the specific scores are not easily comparable with each other. The argument for why the *eyeless* gene was used in this query is that it is longer than the *aniridia* sequence. However it may not matter too much which is used as the template sequence as both results show a high score using local alignments. From the BLAST-search it can also be seen that the E-value (how likely this was found chance) is very low. We can also see that in the alignment 72.59% of the aligned sequences nucleotides are identical, which is very good.

Similar to the discussion in the previous exercise, different scoring parameters could have been used, but since a good result was obtained in this first search, I decided that these parameter-settings worked for this exercise.

2 Chapter 4

2.1 Exercise 2

2.1.1 Background

Hidden Markov Models (HMMs) consists of a few different parts. How these parts come together to form a HMM can be hard to understand from just the equations, matrices and vectors. In this exercise a visual model of a 2-state HMM emitting symbols in the interval [0, 9], its 2 states will be "even" and 'odd".

2.1.2 Methods

Before starting to illustrate the HMM the start vector, transition matrix and the emission probabilities needed to chosen. In this exercise they were just arbitrarily chosen to illustrate the point of how a HMM works, note that there is no real logic behind the specific numbers here, they are just there to represent how a HMM works.

The Transition Matrix, T was defined like this:

Table 1: Transition matrix for an example HMM

	Even	Odd
Even	0.8	0.20
Odd	0.70	

The Emission Matrix, E like this:

Table 2: Emission matrix

	Symbols	0	1	2	3	4	5	6	7	8	9
states											
Even		0.20	0.05	0.10	0.05	0.15	0.05	0.15	0.05	0.15	0.05
Odd		0.10	0.10	0.05	0.15	0.05	0.15	0.05	0.15	0.05	0.15

2.1.3 Results and Discussion

Using the previously described matrices a model of how the associated HMM can be seen in Figure 2.

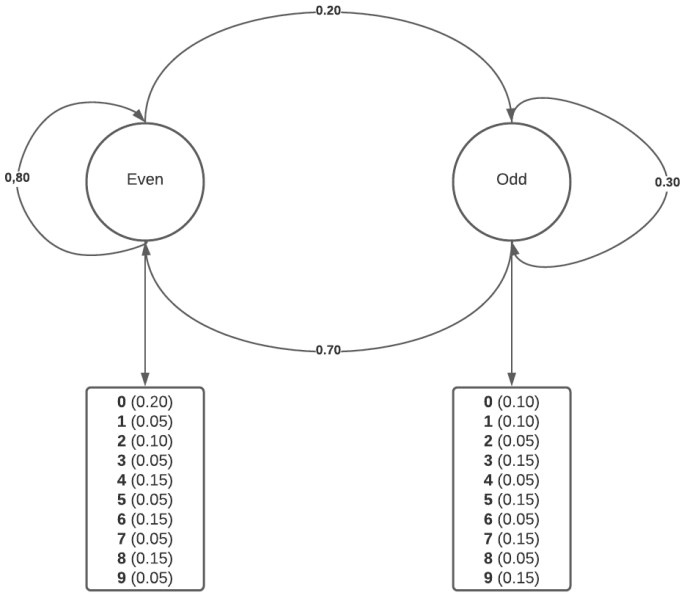

Figure 2: Caption

A few notes could be highlighted from this figure. The arrows going from the circles (states) to either itself or the other state represents T. From the figure it is quite clear that the probability for staying in the same state does not have to be higher than switching state. In this HMM the probability of being in the "Even" state will be very much higher than being in the "Odd" state. The 2 tables below the circles in the figure is represented by E, which then corresponds to the probability that when the model is in a specific state, what probability it has of "emitting" a symbol It can also be noted that just because a state is called "Even" it is not determined that each symbol that fulfills that criteria must be higher than all of symbols that is not associated with that symbol. In this example, the probability of getting a "0" is higher than the other symbols that are even. The probability of getting a "0" in in the state "Odd" is 0.10, which is the same probability of getting a "1". What I am trying to illustrate is that the association between a specific symbol and a state is rather arbitrary. This may be a distinction of small importance however, since in most real and applied HMMs this will most likely be a given, but I thought it was interesting and important to atleast highlight.

3 Chapter 6

3.1 Exercise 1

3.1.1 Background

Comparing homologous sequences from different species can give some interesting information about what types of selection are acting upon said sequence. With the help of the Nei-Gojobori algorithm, previously aligned sequences can compared to estimate the fraction of synonymous (K_S and non-synonymous (K_A) mutations. The ratio R, of K_A and K_S then gives a good hint of what types of selection may be acting on the sequence. It is often said that a R of 1 or higher means that primarily positive selection is acting on the sequence, while a R of lower than 1 is associated with primarily negative selection. It can also be noted that both positive and negative selection is present in a given sequence, then R only gives an estimation of what **primary** selection is acting on the sequence.

In this exercise 3 genes in the mtDNA of human(NC_001807), mouse (NC_005089) and chimpanzee(NC_001643) has been analyzed to see what type of selection is primarily acting on these genes.

3.1.2 Methods

First, all of the sequences were looked up on Genbank, and 3 gene regions as classified by the website were selected (making sure the gene was present in all 3 species as well as seeming to have about the same function). After 3 genes were selected they were exported as FASTA-files and Jalview 2.11.1.7 was used to make a Multiple Sequence Alignment (MSA) of the 3 different sequences (human, mouse and chimp) for each of the genes. Standard settings for doing a MSA in Jalview was used. These MSA's were then exported to FASTA-files.

The MSA FASTA-files were then loaded into the programming language R using the "read.alignment" function in the seqinr package. After all files were loaded the function "kaks" was used to calculate both the K_A and K_S. Finally, R was calculated by: $R = \frac{K_A}{K_S}$ for each of the sequences for all of the genes.

To view the code used for this exercise, see Appendix A.A.

3.1.3 Results and Discussion

The three genes that were selected from the mtDNA's were: *ND1*, *ND2* and *CYTB*. *ND1* can be found at around bp's: $2700 - 3700$, *ND2* can be found at around $3900 - 5000$ and *CYTB* at around $14000 - 15000$.

The resulting R for each of the sequences can be seen in the tables below (rounded to 2 significant digits):

Table 3: R for CYTB

	chimp	human
human	0.14	
mouse	0.11	0.11

Table 4: R for ND1

	chimp	human
human	0.12	
mouse	0.016	0.016

Table 5: R for ND2

	chimp	human
human	0.08	
mouse	0.10	0.13

As can be seen in the tables above, all of the R-values obtained were very, meaning that negative selection is the primary selection on these genes. The gene with the lowest R-values were ND1, between the mouse and chimp and human respectively, which had a R-value of 0.016. The gene with the highest R-value was the CYTB gene between human and chimp.

As previously mentioned, the R-value calculated here is the overall selection acting on the whole gene, it is possible that some regions of the genes has primarily positive selection acting on them, but on the whole, all these mtDNA genes seem to have primarily negative selction acting on them.

If one wanted to capture more of the variation of the selections, it would be possible to utilize a sliding-window plot, to see how the K_A and K_S changed in different regions of the genes. This was not done in this exercise, but could be interesting to investigate in the future.

In this exercise, 3 genes were selected rather arbitrarily, it is possible that other genes do not behave in the same way, in the future if could also be interesting to investigate more of the genes to verify that the results obtained so-fast is consistent. It could also be interesting to investigate other parts of the mtDNA that does not code for a gene. My hypothesis would be that these regions would have a significantly higher R-value than the genes showed, since a mutation here very likely would not be as harmful as in a actually coding region of the DNA.

To conclude, it seems that these 3 genes in the mtDNA of some mammal species have primarily negative selection acting upon them, meaning that they are highly conserved throughout their sequences. I would argue that this makes sense, since mtDNA is so old and "optimized/minimized" to carry out a vital role in the cell (produce ATP, which is the main source of energy in the cells). This would mean that the vast majority of mutations in the important genes in the mitochondrion would be harmful and thus would be selected against.

3.2 Exercise 2

3.2.1 Background

Similar to exercise 1, see section 3.1 for the background to why the analysis of the fractions of synonymous and non-synonymous are may be of interest to figure out what types of selections.

In contrast to Exercise 1, in this exercise we will focus on viral genomes instead of the mtDNA for mammals. More specifically the genome of two different strains of HIV will be analyzed to see what type of selection is primarily acting on their genes. The genomes used in this exercises are: AF033819 and M27323.

3.2.2 Methods

Similar to exercise 1, the sequences were looked up on Genbank and 3 genes were downloaded as FASTA-files from each genome. In this exercise Clustal Omega was used instead of Jalview (standard settings for DNA-sequences were used). The Pairwise Sequence Alignments (PSAs) was exported to fasta files from Clustal Omega.

The PSAs were then loaded into the programming language R using the "read.alignment" function in the seqinr package. After all files were loaded the function "kaks" was used to calculate both the K_A and K_S. Finally R was calculated by: $R = \frac{K_A}{K_S}$.

To view the code used for this exercise, see Appendix A.B

3.2.3 Results and Discussion

The 3 genes that were selected from the HIV genome were: *env*, *gag* and *pol*. *gag* can be found at around bp's: $300-1800$, *pol* can be found at around $1600-4600$ and *env* at around $5800-8300$.

The resulting R for each of the genes can be seen in the table below (rounded to 2 significant digits):

Table 6: R for the genes, gag, pol, env in HIV.

	r
gag	0.38
pol	0.21
env	0.70

The R-values obtained here are all below 1, and thus it is concluded that mostly negative selection are acting on these genes. There is however quite a large difference between their R-values, with the smallest being only 0.21 (*pol*) and the largest being 0.70 (*env*), this means that *env* is quite a bit more variable in its sequence than *pol* is.

Similar to the last exercise, 3 genes were selected rather arbitrarily here, and more genes could be investigated to make sure the results are consistent, and it would also be interesting to compare non-coding regions of the HIV genome to investigate if the R-value is significantly higher there.

Even though all of the obtained R-values are below 1, if one compares them to the R-values obtained from the previous exercise it can quite clearly see that the genes in the viral genome of HIV has far less negative selection than genes in mammal mtDNA. This makes sense, since the HIV is a very rapidly evolving and constantly changing virus, as to be able to avoid being detected by the human immune system, while the mtDNA has a very important function to produce energy for the cells in that organism, so it is more likely that a random mutation would be harmful to the organisms fitness.

4 Chapter 7

4.1 Exercise 1

4.1.1 Background

To determine how closely related different species are to each other, it is common to compare their mitochondrial DNA (mtDNA). To visualize and calculate the relatedness of species, phylogenetic trees are often used. The most common way of calculating trees is by using the Neighbor-Joining Algorithm (NJA). To account to long evolutionary distances between species, some correction is often needed as to not underestimate the number of "hidden" substitutions too much. In this exercise Jukes-Cantor correlation (JCC) will be used, it will also be investigated what effect this has on the resulting phylogenetic tree.

In this exercise the mtDNA of some of the large apes were analyzed and a phylogenetic tree was constructed. The following species was included in this exercise: Gibbon/Hylobates lar (NC_002082.1), Orangutang/Pongo pygmaeus (NC_001646.1), Gorilla/Gorilla gorilla (NC_001645.1), Human/Homo sapiens (NC_001807.4), Neanderthal/Homo sapiens neanderthalensis (NC_011137.1), Bonobo/-Pan paniscus (NC_001644.1) and Chimp/Pan troglodytes (NC_001643.1).

4.1.2 Methods

First, all of the sequences were looked up on Genbank. Using the information on the website, the D-loops were removed from all species which had a D-loop (note that the gibbon did not have a D-loops and thus were left untouched). After that all of the sequences were used to make a MSA, using Clustal Omega with the standard settings for DNA-sequences, the result was then exported to a fasta file.

The file containing the MSA was then loaded into the programming language R using the"read.alignment" function in the seqinr package. After the MSA was loaded the distances between all of the sequences was calculated using the function "dist.alignment" in the same package. To calculate the JCC between the sequences the equation $d_{jc} = -\frac{3}{4}\ln(1 - \frac{4}{3}d)$, where d is the previously calculated distance.

After the two distances were calculated, two trees were made by using a combination of the phylogram and ape packages. First the trees were constructed by using the NJA, then the gibbon node were set to the root (since it is previously known this is the most distantly related to the others). Finally the trees were plotted.

To view the code used for this exercise, see Appendix A.C

4.1.3 Results and Discussion

The resulting trees can be seen in the Figures 3 and 4. It can be seen that they look very similar to each other. The largest difference that can be seen is that the y-axis in Figure 3 has a slightly longer distance between primarily the Gibbon and the other species, while the distance between the others is still very similar.

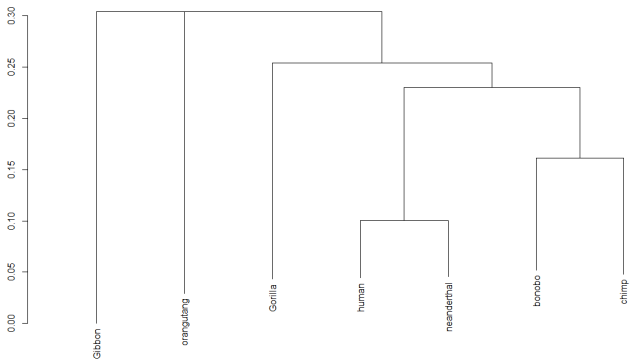

Figure 3: Phylogenetic tree of Gibbon, Orangutang, Gorilla, Human, Neanderthal, Bonobo and chimp using Jukes Cantor Correlation

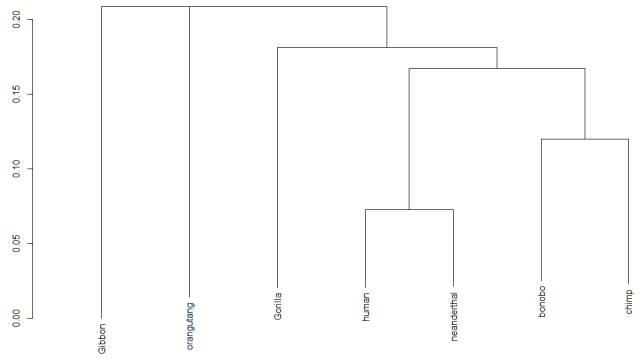

Figure 4: Phylogenetic tree of Gibbon, Orangutang, Gorilla, Human, Neanderthal, Bonobo and chimp without any correlation

The results from both the trees agree with each other to a very high degree, and it can be clearly seen that modern humans and neanderthal are most closely related to each other, and that the chimp and bonobo are share the most recent common ancestor, followed by gorilla, orangutang and gibbon in order. All of these results correspond to other sources I have found on the internet, making me conclude that using distances in the mtDNA of at least apes make it possible to analyze how closely related species are.

It could be interesting to try and keep the variable D-loop in the mtDNA to see if a larger distance would be obtained between the species, and if it would make any significant difference in the analysis.
It would also be interesting to see if the use of other correction methods, like for example Kimura 2-paramter correction, would yield different results than using the Jukes-Cantor correlation.
It would also be interesting to include a few more apes, and potentially other human species to see how the phylogenetic tree would look like then.

Appendices

A R-code

The sections below shows the R-code used to solve the exercises in this report. All code can also be found at this github-link: `https://github.com/AlfonsEdbom/Exercises_Statistics_in_Genetics.git`

A.A Exercise 6.1

The files for the MSA's can also be found in the above github-link.

```
require(pacman)  # Gives a confirmation message.
pacman::p_load(pacman, here, HMM, seqinr, BiocManager, msa)


#load the MSA files (gotten from jalview)
MSA_CYTB_file <- here("sequences", "MSA_CYTB.fasta")
MSA_ND1_file <- here("sequences", "MSA_ND1.fasta")
MSA_ND2_file <- here("sequences", "MSA_ND2.fasta")

#Create alignment object
MSA_CYTB <- read.alignment(MSA_CYTB_file, format = "fasta", forceToLower = FALSE
MSA_ND1 <- read.alignment(MSA_ND1_file, format = "fasta", forceToLower = FALSE)
MSA_ND2 <- read.alignment(MSA_ND2_file, format = "fasta", forceToLower = FALSE)

#Calculate the ka and ks
CYTB_kakas <- kaks(MSA_CYTB)
ND1_kakas <- kaks(MSA_ND1)
ND2_kakas <- kaks(MSA_ND2)
#Calculate the ratio R (ka/ks)
R1 <- numeric(3)
for (i in 1:3) {
  R1[i] <-CYTB_kakas[["ka"]][i] / CYTB_kakas[["ks"]][i]
}

R2 <- numeric(3)
for (i in 1:3) {
  R2[i] <-ND1_kakas[["ka"]][i] / ND1_kakas[["ks"]][i]
}

R3 <- numeric(3)
```

```
for (i in 1:3) {
  R3[i] <-ND2_kakas[["ka"]][i] / ND2_kakas[["ks"]][i]
}
```

A.B Exercise 6.2

```
require(pacman)  # Gives a confirmation message.
pacman::p_load(pacman, here, HMM, seqinr, BiocManager, msa)

#Load the gene files
MSA_env_file <- here("sequences", "Ex6_2", "MSA_env.fasta")
MSA_gag_file <- here("sequences", "Ex6_2", "MSA_gag.fasta")
MSA_pol_file <- here("sequences", "Ex6_2", "MSA_pol.fasta")

#Read the MSA files
MSA_env <- read.alignment(MSA_env_file, format = "fasta", forceToLower = FALSE)
MSA_gag <- read.alignment(MSA_gag_file, format = "fasta", forceToLower = FALSE)
MSA_pol <- read.alignment(MSA_pol_file, format = "fasta", forceToLower = FALSE)

#Calculate the Ka and Ks
env_kakas <- kaks(MSA_env)
gag_kakas <- kaks(MSA_gag)
pol_kakas <- kaks(MSA_pol)

#Calculate R
env_R <-env_kakas[["ka"]][1] / env_kakas[["ks"]][1]



gag_R <-gag_kakas[["ka"]][1] / gag_kakas[["ks"]][1]


pol_R <-pol_kakas[["ka"]][1] / pol_kakas[["ks"]][1]
```

A.C Exercise 7.1

```
require(pacman)  # Gives a confirmation message.
pacman::p_load(pacman, here, HMM, seqinr, BiocManager, msa, phylogram, ape, dnde

#Get the file path to the MSA file
MSA_file <- here("sequences", "Ex7_1", "MSA_mito.fasta")

#Read the file
MSA <- read.alignment(MSA_file, format = "fasta", forceToLower = FALSE)

#Calculate the distance matrix for all the sequences
dist_mat <- dist.alignment(MSA)

#Include the Jukes Cantor correlation for each of the distances
JC_corr <- dist_mat
for (i in 1: length(JC_corr)) {
  JC <- (-0.75*log(1-(4/3)*JC_corr[i]))
  JC_corr[i] = JC
}

#Build neighbour-joining trees
phy1 <- nj(JC_corr)
phy2 <- nj(dist_mat)

#Set the root to Gibbon (known beforehand this is the outgroup)
phy1 <- root(phy1, "Gibbon")
phy2 <- root(phy2, "Gibbon")

#Convert phylo object to dendogram to be able to plot
dnd1 <- as.dendrogram(phy1)
dnd2 <- as.dendrogram(phy2)

#Make into ladder
dnd1 <- ladder(dnd1)
dnd2 <- ladder(dnd2)

#Plot the trees
plot(dnd1)
plot(dnd2)
```