Lab 3: Phylogenetic trees - genes simulated data

BSc Psychobiology, UvA

- ► These are actions for you to do
- ? These are questions
- ⋆ This is a question that could be on the exam

1 Goals

Last week we implemented ~~some~~ computer simulations of evolution in R. This week, we ~~will~~ extend this simulation, and think about the patterns of genetic variation the evolutionary process leaves in a population. We ~~will look at ways in which we can~~ look at methods for *phylogenetic tree reconstruction*. These methods use the genetic variation in the ~~last~~ most recent (current) generation to reconstruct the evolutionary history of a population or a set of species. ~~We will~~

We use a simple clustering algorithm~~for such ‘~~[1] for phylogenetic tree reconstruction~~’ with the goal of understanding~~ . The goal of this lab is to learn about the possibilities and difficulties ~~of approaches based on such algorithms or more complex variants~~that are involved in phylogenetic tree reconstruction.

2 Simulated evolution (continued)

In the previous ~~computer~~ lab, we simulated the evolution of strings of symbols, and looked at the effect of ~~using~~ different fitness functions. Today we will repeat this simulation, but during the evolutionary process we will keep track of ancestry, so that we can reconstruct family trees of different individuals.

- ► Download the scripts for this week's lab and extract ~~them all in the same folder~~ all the scripts to a folder on your computer.
- ► Start R-studio (or ~~a~~ an R terminal) and set ~~your~~ R studio's working directory to the folder ~~you created for the scripts~~ where the scripts reside (if you forgot how, ~~maybe this website~~ have a look at https://support.rstudio.com/hc/en-us/articles/200711843-Working-Directories-and-Workspaces can help you)

We will start with a very small simulation.

- ► If you are working on a university computer, you might have to reinstall the package `stringr`:

```
install.packages("stringr")
```

The ~~file~~ script `lab-3.R` ~~contains a script that~~ runs the same simulation ~~as we did~~ we saw in the previous lab~~, but~~ . This time, however, the script also generates a matrix called `parent_matrix` that ~~stores the evolution process by specifying every point in the simulation~~ specifies the parent of each ~~population member~~ member

[1] An algorithm is a description of a series of steps to do arrive at a certain end result or perform a calculation. Algorithms can for example be implemented by a computer program.

in each generation. Where the parent is the individual of the previous generation whose genetic material was inherited. At the end of the simulation, a plot ~~of~~ illustrating the development of both the average population fitness and the diversity of the population is generated.

- ▶ Change the parameters at the top of the file `lab-3.R`~~.~~ : Set both `population_size` and `simulation_length` to 10.
- ? What values do you expect on the y-axes of these plots?
- ? What do you think the curves of average population fitness and population diversity look like?
- ? At what point do you expect the curves to start and finish?
- ▶ Run the script by executing the following command in the console:

  ```
  source('lab-3.R')
  ```

  ~~Is the result~~
- ? Are the results as you expected?
- ▶ Visualise the parent matrix by running

  ```
  print_parent_matrix(parent_matrix)
  ```
- ? Where in this plot can you find the first generation?
- ⋆ Follow some paths up and down. Why do downward paths often end in dead ends, whereas upward paths always go all the way up?
- ▶ ~~Now change~~ Change the parameters back to ~~its original parameters (~~`population_size` ~~100,~~ `simulation_length` ~~1000), and run~~ their original settings:

  ```
  population_size 100
  simulation_length 1000
  ```
- ▶ Run the simulation again (this may take a while).

Printing the parent matrix for such large simulations is not very helpful ~~anymore~~ (you may try if you want), because the network is too dense to properly visualise. ~~So, rather than looking at it~~Rather than looking the parent matrix, we will use the parent matrix to reconstruct a family tree for only the last generation (that is, we only look at the members of previous generations ~~that have offspring that is still alive).~~whose offspring appears in the last generation.)

- ▶ ~~Generate a tree from~~ From the data you just generated, generate a family tree with the function `reconstruct_tree` and print it with the function `print_tree`:

  ```
  tree <- reconstruct_tree(parent_matrix)
  print_tree(tree)
  ```

  ~~This~~ `print_tree` will generate a ~~string~~ textual representation of the ~~tree, that represents a phylogenetic tree~~To visualise phylogenetic tree.

To generate a visual representation of the tree, we will use an online tree viewer.

- ▶ Copy everything between the double quotes in the output of `print_tree`.
- ▶ Go to http://evolangmus.knownly.net/newick.html. ~~Change~~
- ▶ On the website, change the tree type from *Cladogram* to *Rectangular cladogram*~~and paste~~ .
- ▶ Paste the tree representation you copied into the text area. ~~After clicking~~

- ▶ Click "show". Now, you can zoom in ~~on the tree~~ by scrolling and move ~~it~~ the tree around by dragging ~~it with your~~ the mouse.
- ⋆ As far as you can judge, how many generations ago did the LCA of the current population live?
- ⋆ Which aspects of evolution leave traces that we can detect in the current generation and which aspects do not? ~~Reformulate~~

3 Phylogenetic reconstruction with R

In ~~real life, we usually do not have~~ evolutionary research, the elaborate ancestry information ~~about a population or species~~represented by the parent matrix isusually not available. To reconstruct family trees we have to resort to different methods~~, such as looking at the~~ . Information about when species branched of ('speciated') can be deduced from genetic variation in the current population. For instance, horses and ~~donkeys might genetically be more similar than horses and frogs, so the branches of the latter probably connect further up in the hierarchy than the ones of the former.~~ genetically more similar to donkeys than to, say, frogs. So, the last common ancestor of horses and frogs most likely lived much further in the past than the last common ancestor of horses and donkeys. In other words, the branch that would eventually evolve into frogs split off from the branch that would eventually evolve into horses earlier than the branch that would eventually evolve into donkeys This type of analysis ~~, that is based on a *distance measure* between current population members , is called phylogenetic reconstruction.~~ is called phylogenetic reconstruction. It's based on genetic similarity between members of the current generation, which we measure using a *distance measure*. There are R packages that ~~allow you to automatically do this reconstruction, lets~~ can automatically perform this reconstruction. Lets start with installing these packages:

- ▶ Install the packages `ape` and `phangorn` ~~and load them :~~using:

```
install.packages("ape")
install.packages("phangorn")
```

- ▶ Activate them using:

```
library(ape)
library(phangorn)
```

The `phangorn` package comes with a dataset that contains real genetic data (i.e., RNA samples) from many different species. You can load this dataset by typing:

```
data(Laurasiatherian)
```

To ~~get~~ show a summary of the data you can type `str(Laurasiatherian)`. The data ~~originally comes~~ originates from http://www.allanwilsoncentre.ac.nz/~~, you can have a look there~~ . If you want to find out more about this data, have a look at that website.

~~To reconstruct the evolutionary relationship between the different speciesin this dataset, we start by measuring~~ We will try to reconstruct a phylogenetic tree for these species. That is, we will try to reconstruct when different species branched off from each other, based only on genetic information of the current population (the last generation). The first step is to measure 'genetic distance' between the genetic samples for each species. For simplicity, we assume that all species ultimately originate from ~~one~~ a single common ancestor (an uncontroversial assumption in evolutionary biology), and that species have diverged genetically by picking up mutations at a roughly constant rate (a more problematic assumption).

? (OPTIONAL) If would like to understand this in more depth, try and convince yourself that the phylogenetic tree reconstruction method we described requires the second assumption, and that this assumption is problematic when considering evolution in the real world.

The distance between strings of DNA or RNA is typically measured by counting the number of mutations required to change one into the other. Because of the second assumption, the genetic distance between two species is proportional to the time that has passed since their last common ancestor.

- Select ~~5~~ five species from the Laurasiatherian dataset (for instance ~~3 that~~ three that you think are closely related and ~~2~~ two that are more distantly related) ~~and create~~
- Create a subset of the data containing just these five species using:

```
mysubset <- subset(Laurasiatherian, subset=c(19,20,28,29,30))
```

  ~~(The numbers should~~ The numbers correspond to the position of the species in the list printed by `str(Laurasiatherian)`, i.e. Platypus = 1, Possum = 3, etc.) You have to replace these numbers by the numbers corresponding to the species that you chose.

- Verify that your subset contains the right species using:

```
str(mysubset)
```

- Compute the ~~pairwise distance~~ *pairwise distance* between all elements in the set using the function `dist.ml` and print it

```
distance_matrix <- dist.ml(mysubset)
print(distance_matrix)
```

? ~~Do the numbers~~ The results are stored in a *distance matrix*. How can you read off the distance between two species from this matrix?

? Why are the numbers on the diagonal of this matrix zero?

? Do the computed distances correspond to your intuitions about the selected species' relatedness?

? Using pen and paper, or your favourite drawing software, ~~try to~~ reconstruct a phylogenetic tree ~~(without doing any calculations)~~ that describes the evolutionary relations between your selected species. Use the principles described earlier. You shouldn't need to do any calculations.

We can use a simple method called 'hierarchical clustering' to build such phylogenetic trees automatically. Hierarchical clustering can be done with a simple algorithm that follows these steps:

1. treat each datapoint (for example, a RNA sample of a species) as a separate "cluster" containing just one datapoint;
2. compute the distances between all clusters (using some distance measure; for example, genetic distance);
3. merge the two clusters that are nearest to each other into a new cluster;
4. repeat steps 2 and 3 until only ~~one clusteris left~~all datapoints are in cluster.

To construct a phylogenetic tree, we can ~~represent each merge~~ think of each merging of clusters as the joining of two branches. In the simplest version of this algorithm, we define 'distance' between a cluster A and a cluster B as the average distance between any ~~data point~~ datapoint in A and any ~~data point~~ datapoint in B (a slightly more complicated method, Ward's clustering, uses the square root of the average of the squared point-to-point distances).

⋆ Using the distances between species in `mysubset`, manually perform three cycles of ~~this~~ the hierarchical clustering algorithm with pen and paper.

The `phangorn` package we installed earlier provides ~~several~~ pre-defined functions ~~for~~ implementing different hierarchical clustering methods.

- ▶ Generate a phylogenetic tree for your subset and plot it using:

```
tree <- upgma(distance_matrix, method='average')
plot(tree)
```

- ? Is the tree the same as the one that you created before with pen and paper?
- ▶ ~~Now create~~ Create a tree for the entire dataset.
- ? Does it agree with your expectations? ~~(optional~~
- ▶ (OPTIONAL) Try different ~~ways to compute~~ methods for computing the distance between clusters by changing the parameter `method` (options are, for instance, *ward*, *single* and *median*).
- ? (also OPTIONAL) Do you notice any changes in the resulting phylogenetic trees?

4 Phylogenetic reconstruction of simulated data

We will now investigate what happens if we perform phylogenetic analysis on the population resulting from our own simulated evolution. Remember that, since this is a simulation over which we have full control, we can reconstructed the *actual* phylogenetic tree using the information that we stored in the parent matrix.

- ▶ ~~Rerun~~ Run the script `lab-3.R` again to generate a new population and parent matrix
- ▶ Generate a distance matrix of the last generation from your simulation using the function `compute_distance_matrix`:

```
distance_matrix <- compute_distance_matrix(population)
```

- ▶ ~~Generate a~~ Reconstruct a phylogenetic tree with the upgma function (choose your own *method*) and plot it:

```
tree <- upgma(distance_matrix, method='ward')
plot(tree,cex=0.3)
```

  The parameter *cex* sets the ~~fontsize~~ font size of the plot, adjust it if the numbers are illegible.

- ▶ Now generate the *actual* family tree of the simulation by running

```
gold_standard_tree <- reconstruct_tree(parent_matrix)
print_tree(gold_standard_tree)
```

  and plot it using the online tree visualiser we have used before.

- ? How well ~~did the clustering algorithm reproduce~~ does the reconstruction produced by the hierarchical clustering algorithm match the actual family tree?
- ⋆ How can you explain the differences between the reconstructed and the actual family tree?