



**UNIVERSITÉ
DE GENÈVE**



FACULTY OF SCIENCES
Department of Genetics and
Evolution (GenEv)
Anthropology unit

Computer Skills for Biological Research

*Introduction to GNU/Linux
and bash scripting*

23-27 August 2021

Online session

TEACHERS :

Mathias Currat (mer)
José Manuel Nunes (cc)

EXTERNAL PARTICIPANT:

Jean-Luc Falcone
(University of Geneva)

ASSISTANTS :

Ndeye Faye (as)
Alexandros Tsoupas (as)
Médéric Mouterde (postdoc)
Jérémy Rio (postdoc)
Pascale Gerbault (m. as)
Da Di (m. as)



**UNIVERSITÉ
DE GENÈVE**

FACULTY OF SCIENCES
Department of Genetics
and Evolution

Contents

General introduction.....	5
Data.....	5
Tips.....	6
Evaluation	7
Exercise 0	8
Exercise 1	8
Exercise 2	10
Optional exercise 2a	10
Exercise 3	10
Optional exercise 3a	12
Day 2	13
Exercise 4	13
Exercise 5	13
Exercise 6	13
Exercise 7	14
Day 3	15
Exercise 8	15
Optional exercise 8a	16
Exercise 9	16
Exercise 10	16
Optional exercise 10a	17
Optional exercise 10b	17
Day 4	18
Exercise 11	18
Exercise 12	18
Exercise 13	18
Optional exercise 13a	19
Day 5	20
Exercise 14	20
Exercise 15	20
Optional exercise 15a	20
Teachers and assistants	21
Acknowledgments.....	21
References	22
Data	22
Unix and linux history	22
List of bash commands	22
Programs	22
Survival kit.....	23

Program	Monday August 23	Tuesday August 24	Wednesday August 25	Thursday August 26	Friday August 27
09:15-10:00	Course Introduction GNU/Linux file system, shell, command line utilities (CLI), self-help, pipeline	Recap Day 1 Introduction to sed, grep, and text patterns with escape characters	Recap Day 2 Introduction to variables and for loop Exercise 8: More scripting using variables	Recap Day 3 Using scripts in other languages (R, AWK) Introduction to MDS analysis	Recap Day 4 Introduction to use our computer cluster
10:00-10:15	Coffee break	Coffee break	Coffee break	Coffee break	Coffee break
10:15-11:00	Exercise 0: Using command line utilities (CLI)	Correction exercises Exercise 4: Using sed command and text patterns	Correction exercises (Optional 8a): Using a for loop	Correction exercises Exercise 11: Using an R script	Correction exercises Exercise 14: How to use a GNU/linux computer cluster
11:00-11:15	Coffee break	Coffee break	Coffee break	Coffee break	Coffee break
11:15-12:00	Program installation and introduction to the main research task Exercise 1: Extracting data from pdf file	Exercise 5: Extracting matrices from result files	Introduction to numerical expression Exercise 9: Use numerical expression	Introduction to AMOVA Exercise 12: Automatizing actions	Exercise 15: Running parallel AMOVA using a GNU/linux computer cluster
12:00-13:30	LUNCH BREAK	LUNCH BREAK	LUNCH BREAK	LUNCH BREAK	LUNCH BREAK
13:30-14:15	Introduction to pipeline Exercise 2: Using pipelines (Optional 2a): more pipelines	Introduction to scripting Exercise 6: Making and using bash scripts	Introduction to regular expressions Exercise 10: Using regular expressions & variables	Introduction to parallelism and distributed computing by <i>J.-L. Falcone</i> (CADMOS)	General command recap and Linux history
14:15-14:30	Coffee break	Coffee break	Coffee break	Coffee break	Coffee break
14:30-16:00	Correction exercises Introduction to Arlequin Exercise 3: Create .arp file and use ARLECORE (Optional 3a): Repetition for another locus	Correction exercises Introduction to join, sort and expansions * Exercise 7: Using join, sort and an external script Time to continue the exercises	Correction exercises (Optional 10a): more practice (Optional 10b): more practice Time to continue the exercises	Correction exercises Exercise 13: Launching AMOVA (Optional 13a): Sequential AMOVA Time to continue the exercises	Correction exercises (Optional 16a): Merging results Time for discussion and questions Time to finish the exercises

General introduction

This course intends to present useful computer tools and techniques for researchers in biology, it substantially extends basic computer knowledge acquired at undergraduate levels. Participants will first learn how to use a GNU/Linux workstation environment and how to handle data files (format conversion, extraction of data, etc...) using GNU/Linux bash commands and simple shell scripts [1]. The use of scripts to facilitate multiple analyses and the compilation of numerous data gathered from different sources will be exemplified throughout the five days of the course. The participants will have to develop, incrementally, the command lines and the scripts that will allow them to solve the exercises proposed. Upon conclusion of the course, participants are expected to know how to use comfortably a GNU/Linux environment with its basic commands and find adequate solutions to handle data and conduct data analyses in the context of their research.

Theoretical bases will be given to introduce all aspects encountered by the participants during the course. However, the main point of the course is practical training. Most exercises will be done in the context of a specific problem of population genetic data analysis: evaluate the existence of genetic differentiation between human populations located in each side of the strait of Gibraltar using a database of forensic markers made up of 15 STR (short tandem repeat).

During the first day, participants will get acquainted with the environment and will create a file in a format suitable to the program ARLEQUIN [2,3] to perform an analysis of genetic distances between populations from a genetic dataset recovered from a .pdf file retrieved from a data folder. The second day will lead to the use of more efficient commands and to basic scripts. The third day will lead to the use of scripting as a means to automate data formatting and conversion and to a more efficient use of regular expressions. The fourth day will show how to combine scripts written in different languages such as AWK [4] or R [5]. The fifth day will address parallelisation and the use of computer clusters. During the last afternoon, participants will have time to finish and discuss their exercises and ask questions related to their own needs.

Data

During this training course, the participants will handle and analyze a dataset of genetic data typed in human populations from each side of the Gibraltar Strait. Those data are made up of 15 STR (Short Tandem Repeat or microsatellite) obtained in laboratories for samples of unrelated individuals living in various cities or regions. Those specific 15 genetic loci are commonly used in forensic studies, namely for the purpose of paternity testing, and have been published in specialized scientific journals.

The participants will retrieve the data from the corresponding scientific articles in .pdf format [6-8] and then extract, combine and analyze them using ARLEQUIN and R. Some analyses will be done by including data taken from the online database www.strdna-db.org [9].

All the input files required for each exercise will be provided in the corresponding folder located in the folder CSBR/Exercises_data. More indications to get those files will be given during the course.

For each exercise, we recommend that you start by using the files provided in the server instead of your own files from previous exercises, to avoid passing on errors that may have occurred and to facilitate corrections. Once it is working with the input files provided, then you can try with you own files.

Tips

- Exercises are subdivided into successive steps. At each step, save the current data into a file to which you will give a specific name (the proposed filename appears in bold within squared brackets and the end of each step, such as **[filename]**). You may use more steps if necessary, but it would be easier to follow the course if you stick to those proposed. You may want to use temporary files to keep intermediary steps of your work. We suggest that you use numeric extensions (.1,.2, etc...) to easily identify such files. Final files should have more meaningful, typically alphabetic, extensions, such as .txt, .list, .arp, etc...) — this will be the case if you follow our names **[suggestions]**. The temporary files are important to go back to a particular step if necessary, so don't delete them! — Before the end of the course you will learn how to delete those files really quickly!
- You should test various commands for every steps of the exercise, but once you have found the appropriate command for a given step, please copy it into a file that you will keep updated (you may use a file for each exercise). At the end of each exercise, you should be able to repeat all steps again using your file. Do not forget to include comments for every command.
- At each step of an exercise, some commands are proposed to realise the task. Of course, there are always many other ways to perform a given task, but try to use those that are suggested as it will be much easier to follow the corrections thereafter. Note that in most cases you will need to use more commands than those mentioned to realise the task (only the most important or new commands are mentioned at each step).
- You will find a “survival kit” at the end of the current textbook, where the most useful commands are listed.

Evaluation

In order to credit your participation to the course:

1. **Your active “online presence” is mandatory during the 5 whole days !**
2. You will need to **write a personal report** and send it by email to mathias.currat@unige.ch before October 1st 2021.
3. If you belong to the University of Geneva, you will have to **pass a short oral exam** at a date to be arranged once the report has been received. The purpose of the oral examination is to discuss the report as well as an additional script that you will receive a few days before the exam. You will have to explain the script to the teachers. If you belong to another University and participate to the course through the CUSO DPEE program, then this discussion is optional.

Your report will consist in scripts created during the course with your own comments explaining the goals of the script and their successive steps. In addition, you must include in the beginning of your report a personal and general summary of the course (one page). This should present your work during the course as well as the results obtained and, possibly, include general comments that you may have about the usefulness of using bash scripting and scripting for your future research work. The whole report must be sent as a single pdf file with your name in the filename.

Please note that **the report needs to be personal for each participant**, even if you collaborated with other students during the course. Reports which are too similar to each other or which consist in a simple series of copy/paste from the textbook and/or teacher's presentations will not be considered and the participation to the course will not be validated!

Day 1

During this first day, you will first learn how to get comfortable with the Ubuntu graphical environment. Then, you will learn how to use command lines (CLI) in practising very useful commands, which will constitute the basis for the rest of the course. You will use CLI to launch an analysis of genetic diversity using the linux version of the program ARLEQUIN.

Exercise 0

The exercises are included in the hand-outs of the presentation and are expected to be done by the participants in “echo” to the presentation. They consist of commands that make the “survival kit” of the shell and command line utilities. They emphasize the regularities expected within a shell environment such as options, arguments and redirections, working directory and paths.

Exercise 1

In this exercise you are expected to retrieve a file from a folder CSBR/Exercises_data and extract the data table of interest from that .pdf file. In order to have an overview of your work, please read all the exercise before proceeding to execute commands.

In the folder in your computer where you would like to put all exercises of the course, please type the command “mkdir Exercise01” and press return. What have you done?

Now type the command “cd Exercise01” and press return. What have you done?

If you are unsure about the meaning of the two commands above, you can read their manual by typing either “**man** cp” or “man mkdir”.

Now that you have created a working folder in your computer for exercise 1, you need to recover the file Coudray_et_al_ForSciInt_07.pdf from the folder CSBR/Exercises_data/Ex01/ and save a copy into your working folder using the command **cp**. Use the manual to find the correct command.

You can now convert the .pdf file to a text (.txt) file using the command **pdftotext**. In order to make your work easier you should check which options are available for this command and their effects in the output. For instance, you can use an option to select the first page to convert and another one to select the last page to convert and finally an option to keep the table’s structure in the output. As an example, the structure of the command should be something like “pdftotext -option1 -option2 -option3 filename.pdf”). You can find these options by consulting the command manual, which can be reached by typing “pdftotext -h” or “**man** pdftotext”. You may also want to start by seeing the output in the screen instead of directing it to a file, for this use – in place of the output filename (– means

standard output, stdout, normally the terminal screen). The goal is to get the table 1 (allele frequency) in text format. Note that you do not need the last 5 lines of the table (those starting with “Ho”).

[pdf_table.data]

Once you have the table in a text file, possibly with some other text before and after it, you may use the commands **head** and **tail** to select only the required lines (or to eliminate the unwanted ones) from the text file. You should also use the command **cat**.

[table.data]

Allele	D3S1358	D21S11	D18S51	D5S818	D13S317	D7S820	D16S539	CSF1PO	D8S1179	FGA
6	-	-	-	-	-	-	-	-	-	-
7	-	-	-	-	-	-	-	0.018	-	-
8	-	-	-	-	0.004	0.175	0.171	0.018	0.009	0.009
9	-	-	-	-	0.044	0.057	0.105	0.105	0.022	0.022
9.3	-	-	-	-	-	-	-	-	-	-
10	-	-	-	0.018	0.026	0.075	0.303	0.053	0.338	0.338
11	-	-	-	0.026	0.377	0.320	0.232	0.329	0.276	0.276
12	-	-	-	0.171	0.355	0.219	0.118	0.303	0.276	0.276
13	-	-	-	0.123	0.180	0.118	0.044	0.158	0.057	0.057
13.2	-	-	-	-	-	-	-	-	-	-
14	0.088	-	-	0.180	0.013	0.031	0.009	0.031	0.022	0.022
14.2	-	-	-	-	-	-	-	-	-	-
15	0.189	-	-	0.167	-	0.004	-	0.004	-	-
15.2	-	-	-	-	-	-	-	-	-	-
16	0.206	-	-	0.136	-	-	-	-	-	-
16.2	-	-	-	-	-	-	-	-	-	-
17	0.241	-	-	0.061	-	-	-	-	-	-
18	0.211	-	-	0.061	-	-	-	-	-	-
19	0.061	-	-	0.026	-	-	-	-	-	-
20	0.004	-	-	0.022	-	-	-	-	-	-
21	-	-	-	0.009	-	-	-	-	-	-
22	-	-	-	-	-	-	-	-	-	-
23	-	-	-	-	-	-	-	-	-	-
23.2	-	-	-	-	-	-	-	-	-	-
24	-	-	-	-	-	-	-	-	-	-
25	-	-	-	-	-	-	-	-	-	-
25.2	-	0.004	-	-	-	-	-	-	-	-
26	-	-	-	-	-	-	-	-	-	-
27	-	0.035	-	-	-	-	-	-	-	-
28	-	0.162	-	-	-	-	-	-	-	-
29	-	0.228	-	-	-	-	-	-	-	-
30	-	0.197	-	-	-	-	-	-	-	-
30.2	-	0.057	-	-	-	-	-	-	-	-
31	-	0.101	-	-	-	-	-	-	-	-
31.2	-	0.079	-	-	-	-	-	-	-	-
32.2	-	0.083	-	-	-	-	-	-	-	-
33	-	0.009	-	-	-	-	-	-	-	-
33.2	-	0.035	-	-	-	-	-	-	-	-
34.2	-	0.009	-	-	-	-	-	-	-	-

Figure 1. Example of “table.data” file

You may use **gedit** (a text editor) to see your file if you wish. You may also make changes using this editor, but this is not wise because you will not be able to automate them, so if you want to make some changes, prefer using command lines.

Now that your table is in a text format, check that you have written out all the commands that you have used to generate “table.data” from the file “Coudray_et_al_ForSciInt_07.pdf” in a file that will be used for your final report. You should add comments to explain each of your CLI (anything from a # sign to the end of the line). Do not forget to remove temporary files at the end.

[ex01.txt]

Exercise 2

Modify your series of command in ex1.txt by using pipes “|” in order to avoid creating too many temporary files. The series of command must still generate “table.data” from the file “Coudray_et_al_ForSciInt_07.pdf”.

[ex02.txt]

Optional exercise 2a

Repeat exercise 2 but do it in one single step, without any temporary file. Use the sed command. Compare the resulting table and the one in the pdf to check that all the lines are present.

[opt02a.txt]

Exercise 3

Now that you have a plain text file with the frequencies table, you will create a text file in ARLEQUIN format (.arp file) containing the data extracted during exercise 1 for locus D3S1358 and additional samples. You will validate this file by calculating genetic distances using the program ARLECORE (Linux version of the program ARLEQUIN).

First, you need to get the data concerning the locus of interest. Use the command **cut** that allows to select columns from a plain text file; do not forget to check its options to see how to specify where and how to cut. It is suggested that you start by seeing the output of your commands in the stdout (the screen terminal) and then directing it to a text file, using the > redirection operator. Of course, you can always use **gedit** to check the result.

[all_alleles.data]

Now that you have only the D3S1358 locus data in a file, you are going to remove the lines that correspond to absent alleles (no frequency values). Use **grep** with the invert-match option. Once you are satisfied with the output, redirect it to a file.

[significant_alleles_locus_1.data]

Then create a file containing the new sample (Huelva-spain) in the ARLEQUIN format. Use the file “dataset_D3S1358.arp” as example. Get the data from your file [significant_alleles_locus_1.data]. It should be as the example below, starting with three keywords indicating the sample name, the sample size (in number of gene copies, that you can retrieve by reading the initial pdf file) and finally the sample data, which list each allele and its frequency in the population sample.

```

SampleName="Huelva-Spain"
SampleSize=228
SampleData={
    14    0.088
    15    0.189
    16    0.206
    17    0.241
    18    0.211
    19    0.061
    20    0.004
}

```

[new_sample.txt]

Now you have to create the header for the ARLEQUIN file. Write it using the command **echo** (or recover it from the folder CSBR/Exercises_data) an arlecore header into a text file. You can use the header of file “dataset_D3S1358.arp” as example, but you need to increase the number of samples (from 20 to 21) because you will add the sample from Huelva to the current dataset made up of 20 population samples. It should be similar to the following example:

```

[Profile]
    Title="All samples at Gibraltar for locus D3S1358"
    NbSamples=21
    DataType=FREQUENCY
    GenotypicData=0
    Frequency=REL
[Data]
[[Samples]]

```

[header.txt]

You should now merge the header, the new sample and all the remaining samples from dataset_D3S1358.arp into one single ARLEQUIN file (.arp). Instead of copy/paste using **gedit**, use the command **cat** and the redirection operators **>** and **>>** to create the .arp file. The file should thus contains the header with the correct number of population samples, then list all population samples one after the other below the keyword “[[Samples]]”. Please note that the format of Arlequin is very strict.

[locus_D3S1358.arp]

From the folder CSBR/Exercises_data recover the settings file FstSettings.ars required to run ARLECORE and put it in the same directory or folder than the .arp file. By running ARLECORE you will check that the .arp file you have made is correct. Check that your working directory is where the .arp and .ars files are located (use **pwd**) and if necessary change to it (using **cd**), then type **arlecore** followed by the name of the .arp file and the name of the settings file (FstSettings.ars). See the

results in the .res directory. You should find a matrix of genetic distances between each pair of population samples among other results, in the .xml file.

Write all your commands for exercise 3 into a file. You may use the command **history** to help you.

[ex03.txt]

Optional exercise 3a

Merge all the successful commands from exercises 1 and 2 into one single file and add a series of new commands to generate an ARLEQUIN file for the second locus (D21S11). All those commands taken together should extract the file table.data and then generate two .arp files: one for the locus 1 and one for the locus 2.

[opt03a.txt]

Day 2

Today the actual benefits of working with CLI and text files will become much more impressive to you. You will first extract, manipulate and sort data. Then, you are going to apply your series of commands to other loci automatically. Just follow!

Exercise 4

Improve exercise 3 in extracting the header for the ARLEQUIN file using the **sed** command and text patterns.

Exercise 5

During the exercise, you will extract two matrices of results and one list of labels among all the results generated by ARLECORE during exercise 4 (or 3).

First of all, copy the result folder “locus_D3S1358.res” obtained during exercise 4 (or 3) into your working folder. In the file “locus_D3S1358.xml”, there is a matrix of coancestry coefficients, containing the genetic distances between all pairs of population samples. Extract the content of this low triangle matrix into a new file (only the distance values without any header or legend). Use the series of command that you already learned, such as **cat**, **cut**, **head**, **tail**, **sed** as well as **tr**.

[locus_D3S1358.dist]

Then, apply the same strategy for extracting the matrix of significant *Fst* *P* values. The final square matrix contains + and – signs instead of numerical values.

[locus_D3S1358.sign]

Finally, extract the list of population’s labels. It should start with first label and end with the last one.

[locus_D3S1358.lab]

Copy all your successful command into a file!

[ex05.txt]

Exercise 6

You will now create your first bash script.

First, put the series of commands of exercises 4 into a single file and transform this file into a bash script. Run it to check that you obtain the same **locus_D3S1358.arp** file that you got for exercise 4.

[createarpfile.sh]

Secondly, do the same with the series of commands of exercise 5 and check that you obtain good .dist, .lab and .sign files with this script.

[extractmatrixes.sh]

Finally, create one new bash script that executes createarpfile.sh and extractmatrixes.sh successively. You should have the files table.data and FstSettings.ars as input files in your working folder.

[ex06.sh]

Exercise 7

We are now providing you with a script called serialise_arp_tables.sh. You need to apply this script to both matrix files generated during exercise 6 (locus_D3S1358.dist and locus_D3S1358.sign) in calling it with their generic name (without the file extensions) as parameter.

First execute the script “serialise_arp_tables.sh” in your working folder.

Once the script has run successfully, rename the results files to

[dist.1]

[sign.1]

From the file **sign.1**, extract only the lines for which the distance is statistically significant “+” into a new file.

[sign.2]

Finally, using the command **join**, merge the two files in order to get only the pairs for which the distance is significant, but ordered by the distance (from the largest to the smallest). This file should contain two columns: the population pairs and the distance value (knowing that they are all significant). Do not forget to **sort** your files before joining them.

[difpop_D3S1358.txt]

Answer to those two questions:

1/ what is the number of population pair which are genetically differentiated?

2/ which are the two populations the most differentiated genetically?

Write all your successful commands for exercise 7 in one script in order to be able to reproduce this exercise easily. Check that it works!

[computediffpop.sh]

Day 3

During day 3, you will improve your knowledge of bash scripting through the use of variables, numerical and regular expressions, loops and conditionals.

Exercise 8

Now that making a one—locus file is done, let us proceed to two other loci.

Copy your scripts createarpfile.sh, extractmatrixes.sh and computediffpop.sh to your working folder and rename them createarpfile_D3S1358.sh, extractmatrixes_D3S1358.sh and computediffpop_D3S1358.sh. Then create a new script “make_locus_D3S1358_arp_file.sh” which calls the three former scripts successively. Also copy the input files table.data, serialise_arp_tables.sh and FstSettings.ars in your working folder.

[make_locus_D3S1358_arp_file.sh]

Now create a new series of scripts in replacing D3S1358 by D21S11 in the filenames. Make the modifications required on those new scripts to obtain an .arp file for the second locus D21S11 and run it.

[make_locus_D21S11_arp_file.sh]

Now create a new series of scripts in replacing D21S11 by D18S51 in the filenames. Make the modifications required on those new scripts to obtain an .arp file for the third locus D18S51 and run it.

[make_locus_D18S51_arp_file.sh]

Now list the differences between the scripts for the three loci and create a series of generic scripts, using at least three arguments (e.g., the **\$1**, **\$2** and **\$3** parameter) for the parts that are changing among scripts (e.g., the locus number, the number of populations before and after adding Huelva and the number of the column that holds the locus data). In addition, can you make your script change the name of the final output file according to the name of the locus (and not to its number)?

[make_one_locus_arp_file.sh]
[computediffpop.sh]
[createarpfile.sh]
[extractmatrixes.sh]

Generate an .arp file for all the loci using your last generic script.

You will find all the datasets necessary of the exercise in the corresponding folder in CSBR/Exercises_data. Note that all input file do not exactly contain the same number of populations, so you may need to adapt your script in order to account for those differences.

Please do not forget to annotate your scripts with all the comments you think of interest and also other commands you may have used; you will certainly find them useful for the final report.

Optional exercise 8a

Extend the procedure of exercise 8 using a **for** loop, in order to obtain .lab, .dist and .sign files for every loci thus making the same analysis for all loci using only one single command!

[opt08a.sh]

Exercise 9

Get the last version of your script createarpfile.sh created for exercise 8 and improve it in extracting the number of populations automatically instead of passing it as parameter.

[createarpfilebest.sh]

Exercise 10

In this exercise, you are going to practice regular expression and to understand how useful it is.

First, go back to your last version of your script createarpfilebest.sh 9 (exercise 9) and replace the command tr which remove series of space by the command **sed** and a regular expression, in order to remove series of blank by a tabulation.

[createarpfile.sh]

Then go back to all the scripts of exercise 8 and make all the possible changes using regular expressions and command such as sed or grep in order to generate proper files .dist, .sign and .lab. You will also improve the scripts in introducing local variables when this is useful, i.e. when a file name is used at several places in the script. Your scripts should use the correct population number, depending on the locus under consideration.

[make_all_loci_arp_file.sh]

[improved_createarpfile.sh]

[improved_extractmatrixes.sh]

[improved_computediffpop.sh]

Optional exercise 10a

Exciting new data just arrived (population sample from Egypt and Morocco) in the form of two new pdf files, you can find them in CSBR/Exercises_data/ in the data directory for optional exercise 10a. Modify your script **improved_createarpfile.sh** from exercise 10, in order to extract the table for each new sample from the pdf. Your script should also extract the sample size. You should have one generic script, which takes as input parameter the pdf filename (and maybe more parameters), working for all sample.

[make_table_data.sh]

Then, create a new script called make_arp_files_from_table_data.sh, which create one arp file for each sample and each locus.

[make_arp_files_from_table_data.sh]

Finally, create another script that merges the new sample arp file with the arp file with all samples that you have generated during exercise 9. This script should be applied independently to each locus.

[merge_arp_files_for_locus]

[opt10a.sh]

Optional exercise 10b

The goal of this exercise is to identify the pairs of populations that are the most often genetically differentiated. For this, get all the diffpop_*.txt files and the *.lab files generated during exercise 10.

The first thing that you need to do is to replace the population numbers in the diffpop_*.txt files by the sample names from the corresponding *.lab files because there could be different number of populations for different loci.

Then you may change the order of the two populations in each file in order to always have the first alphabetical one at the beginning of the line. It would avoid getting the same pair written differently (e.g. p1~p2 or p2~p1).

[names_"locusx".pairs]

Finally, count the occurrence of each pair among all loci using the command **uniq** and sort it from the most common to the less common.

[pairs.count]

You can also list the number of loci for which each sample is available in computing their occurrence among the *.lab files.

[samples.count]

[opt10b.sh]

Day 4

During day 4, you will learn more bash commands and how to use scripts written in a different language (here R and AWK) within bash.

Exercise 11

Scripting will be used with the statistical package R in order to produce MDS analyses. Recover the distance matrix and the labels for the locus CSF1PO generated during exercise 10 (.dist and .lab files) as well as the MDS R script from the folder CSBR/Exercises_data. First, add the sample names from the .lab file as a first header line to the .dist file and put everything in a new file. Use the command **paste**.

[locus_CSF1PO.raw]

Then, replace -0.0000 values in the matrix by 0.0 values.

[locus_CSF1PO.mat]

Then perform a MDS using the R script.

[ex11.sh]

Exercise 12

Automatize the creation of MDS for all loci.

[ex12.sh]

Exercise 13

Use an “**awk**” script (MakeStructure.awk available in folder CSBR/Exercises_data) to append a “population structure” to the arp file containing all populations for the locus CSF1PO (locus_CSF1PO.arp, generated during exercise 10). The population structure must be constituted of two groups: one with all Africans populations and the other one with all European populations. First, create a file with the structure only using the awk script. Second, create a new file that contains both the content of the initial arp file AND the structure at the end.

[locus_CSF1PO_struct.arp]

Launch an AMOVA analysis on the arp file produced to check if its format is valid, use the settings file for ARLEQUIN called AmovaSettings.ars.

Optional exercise 13a

Modify your previous script done in exercise 13 in order to generate new arp files with a population structure grouping all African population in one side and all European populations in the other side, for all loci.

[create_arlfilesAMOVA.sh]

Then create a second script that computes an AMOVA for all ARLEQUIN files (.arp) with a population structure.

[launch_AMOVA.sh]

Finally, create a script that calls both previous scripts successively.

[opt13a.sh]

Day 5

During this last day, you will also learn how to use a Linux computer cluster. You will also have time to finish exercises from the previous days.

Exercise 14

Now you will launch the computation of AMOVA for one locus (choose the one you want) in the cluster. Use the command `sbatch` to launch a script which, 1/ create a working for that locus, 2/ copy all the needed files in it, 3/ launch the AMOVA computation using ARLECORE.

[launch_AMOVA.sh]
[ex14.sh]

Exercise 15

Now that you have launched the AMOVA for one locus in the cluster (exercise 14), and maybe sequentially for each locus on your local computer (optional exercise 13a), you will launch AMOVA in parallel for every loci on the cluster. The AMOVA analysis for each locus must be sent independently to each node of the cluster using the command **sbatch**.

Before launching the analysis, please change the number of permutations from 1,000 to 100,000 in the file "AmovaSettings.ars" at line "NumPermutationsAMOVA=".

Your script launching the AMOVA for one locus should first create one directory for that locus and copy all needed files.

Finally retrieve only the XML (.xml) results file for each locus and copy them in a folder called AMOVARESULTS.

[launch_AMOVA_improved.sh]
[ex15.sh]

Optional exercise 15a

After having launched AMOVA for each locus on the cluster, make a script that retrieve the results (Fixation indices *Fst*, *Fct* and *Fsc* values, as well as their respective *P-values*) in one single summary file. Use of **sed** may help you for this exercise.

[ex15a.sh]

Teachers and assistants

Mathias Currat: mathias.currat@unige.ch

José Nunes: jose.deabreununes@unige.ch

Pascale Gerbault: pascale.gerbault@unige.ch

Da Di: da.di@unige.ch

Jérémy Rio: jeremy.rio@unige.ch

Médéric Mouterde : mederic.mouterde@unige.ch

Ndeye Faye : ndeye.faye@unige.ch

Alexandros Tsoupas: alexandros.tsoupas@unige.ch

Acknowledgments


Computer facilities and organization:

Stephan Weber (UA-GENEV, University of Geneva)

Caroline Stemberger Duri (UA-GENEV, University of Geneva)

Marta Bellone (CUSO)

Support:

University of Geneva,  Department of Genetics and Evolution - Unit of Anthropology



Conférence Universitaire de Suisse Occidentale (CUSO)

Doctoral Program in Ecology and Evolution



References

1. bash <https://www.gnu.org/software/bash/manual/bashref.html>
2. Schneider S, Kueffer J-M, Roessli D, Excoffier L (1997) Arlequin "A Software for Population Genetic Data Analysis". 1.1 ed. Geneva: Genetics and Biometry Laboratory Dept. of Anthropology and Ecology University of Geneva.
3. Excoffier, L. and H. E. Lischer (2010). "Arlequin suite ver 3.5: a new series of programs to perform population genetics analyses under Linux and Windows." *Molecular Ecology Resources* 10(3): 564-567.
4. awk <https://www.gnu.org/software/gawk/manual/gawk.html>
5. R Development Core Team (2010). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL <http://www.R-project.org>.

Data

6. Coudray, C., E. Guitard, et al. (2006). "Allele frequencies of 15 short tandem repeats (STRs) in three Egyptian populations of different ethnic groups." *Forensic Sci Int* 169(2-3): 260-265.
7. Coudray, C., E. Guitard, et al. (2007). "Population genetic data of 15 tetrameric short tandem repeats (STRs) in Berbers from Morocco." *Forensic Sci Int* 167(1): 81-86.
8. Coudray, C., R. Calderon, et al. (2007). "Allele frequencies of 15 tetrameric short tandem repeats (STRs) in Andalusians from Huelva (Spain)." *Forensic Sci Int* 168(2-3): e21-24.
9. Pamplona, J.P., F. Freitas, L. Pereira (2008). "A worldwide database of autosomal markers used by the forensic community." *Forensic Sci Int: Genetics Supplement Series* 1: 656–657

Unix and linux history

<http://www.computerhope.com/history/unix.htm>
http://en.wikipedia.org/wiki/History_of_Linux

GNU software manuals

<http://www.gnu.org/manual/>

List of bash commands

<http://ss64.com/bash/>

Programs

ARLEQUIN: <http://popgen.unibe.ch/software/arlequin35>

R: <http://cran.r-project.org/> (but you can install it easily from your distribution)

Survival kit

```
>          ## redirection of output
>>        ## append output
|          ## pipe
||         ## logical or
;          ## end of statement
.          ## current directory
..         ## parent directory
(expr)     ## evaluate arithmetic expr
[[expr]]   ## test logical expr
$1         ## positional parameter 1 (can go up to 9)
cat        ## concatenate and display contents on stdout
cd         ## change directory
cp         ## copy file(s)
csplit     ## split file
cut        ## retrieve selected columns
do while expr ## loop while expr is TRUE
done       ## close loop (for or while)
echo       ## send argument to stdout
for list; do ## loop for each element of list
grep       ## select lines based on pattern
head       ## retrieve initial lines
if then fi ## conditional branch
join       ## combine common lines of two files
ls         ## list current directory
mkdir      ## create directory
mv         ## move, or rename, file(s)
paste      ## merge lines from file(s)
pdftotext  ## convert pdf to text
rm         ## remove file(s)
Rscript    ## execute the script argument in R
sbatch     ## put script argument on the cluster batch queue
sed        ## perform line by line manipulation
seq        ## create regular sequence
sort       ## order lines of file
source     ## execute argument as a bash script
tail       ## retrieve last lines
test       ## test logical expression
tr         ## translate characters
uniq       ## retrieve unique values
wc         ## statistics of lines, words and characters
```