# Graduate School - Unix
# Sequence analysis on a Linux system

Barbera DC van Schaik
Bioinformatics Laboratory, KEBB
Amsterdam UMC
b.d.vanschaik@amsterdamumc.nl

March 5, 2020

In this document we assume that you are familiar with terms like: fasta, blast or blat, sequence alignment and assembly. When this is not the case don't hesitate to ask me for further information.

The linux commands in this document that you use for sequence analysis are in the boxes. Indented lines belong to the same command and should be typed on the same line (so without pressing "enter" in between).

**Do not copy and paste from this document unless it is explicitly mentioned!!** a) Copy/paste often doesn't work. b) You will remember the commands better if you type it in yourself.

## Contents

# 1 Login to the server

For this practical we use the HPC cloud system of SurfSara `https://surfsara.nl/`. We will use a lightweight shell client, Putty (`http://www.putty.org/`), to connect to the server. In addition to this we will need an X server for graphical display (Xming, `https://sourceforge.net/projects/xming/`)

1. Start the X server and then Putty. Configure Putty as follows:
2. Fill in the ip-address in the "hostname" box
3. Select SSH >X11 in the menu and enable X11 forwarding
4. You can save the settings in the "Session" menu
5. Then select "Open" and login with username and the password that was handed out.

Alternative: When you have Linux or Mac you can login like this.

```
ssh −X studentNN@145.100.x.y
```

Send an e-mail to Barbera (b.d.vanschaik@amc.uva.nl) when you experience problems with your virtual machine

# 2 A quick example

In this example you will count the nucleotides in a sequence. Create a new file with the "nano" editor and copy/paste the sequence below in the editor.

```
nano HBA1.fasta
```

```
ACTCTTCTGGTCCCCACAGACTCAGAGAGAACCCACCATGGTGCTGTCTCCTGCCG
ACAAGACCAACGTCAAGGCCGCCTGGGGTAAGGTCGGCGCGCACGCTGGCGAGTAT
GGTGCGGAGGCCCTGGAGAGGATGTTCCTGTCCTTCCCCACCACCAAGACCTACTT
CCCGCACTTCGACCTGAGCCACGGCTCTGCCCAGGTTAAGGGCCACGGCAAGAAGG
TGGCCGACGCGCTGACCAACGCCGTGGCGCACGTGGACGACATGCCCAACGCGCTG
TCCGCCCTGAGCGACCTGCACGCGCACAAGCTTCGGGTGGACCCGGTCAACTTCAA
GCTCCTAAGCCACTGCCTGCTGGTGACCCTGGCCGCCCACCTCCCCGCCGAGTTCA
CCCCTGCGGTGCACGCCTCCCTGGACAAGTTCCTGGCTTCTGTGAGCACCGTGCTG
ACCTCCAAATACCGTTAAGCTGGAGCCTCGGTGGCCATGCTTCTTGCCCCTTGGGC
CTCCCCCCAGCCCCTCCTCCCCTTCCTGCACCCGTACCCCCGTGGTCTTTGAATAA
AGTCTGAGTGGGCGGC
```

In the menu below you see the short cuts for the editor. With control+x you will exit the editor, it will ask you to save the changes. Check if the file was generated correctly with

```
cat HBA1.fasta
```

The following one-line program counts each nucleotide. If you would like to see what happens in each step you can execute the parts one by one.

```
cat HBA1.fasta
cat HBA1.fasta | sed 's/.\{1\}/&\n/g'
cat HBA1.fasta | sed 's/.\{1\}/&\n/g' | sort
cat HBA1.fasta | sed 's/.\{1\}/&\n/g' | sort |uniq -c
```

# 3 Download and install programs

Make a sub directory called "programs" and one with the name "data". Go to the directory "programs" and download and install the BLAST program.

```
wget ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/ncbi-
    blast-2.10.0+-x64-linux.tar.gz
tar zxvf ncbi-blast-2.10.0+-x64-linux.tar.gz
export PATH=${PATH}:${PWD}/ncbi-blast-2.10.0+/bin
```

You can copy/paste the url from here: `ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/ncbi-blast-2.10.0+-x64-linux.tar.gz`. With wget you can download files. In this case the file is downloaded from the FTP server at the NCBI. In a similar way you can download files from regular web pages (http or https). The tar program is used to extract tar archives. The z option unzips the archive, x extracts the tar archive, v stands for "verbose", and f forces the program to overwrite files if they already existed. With the last command the BLAST programs are added to a variable called "PATH". This means that in whatever directory you are you can type the name of the program without having to specify the entire path to the executable. Go to your home directory (cd) and type

```
blastn
```

to try this out. You should get a message like this: "BLAST query/options error: Either a BLAST database or subject sequence(s) must be specified Please refer to the BLAST+ user manual."

# 4 Getting data from the web

For this exercise we will use a metagenomics sample that was collected at a glacier in Germany [1]. You can read more about the project here: `https://www.ebi.ac.uk/metagenomics/project/SRP000240` In most cases, sequences from a metagenomics study will be assembled first and then matched against public databases. We will skip the assembly step.

First download the 16S Ribosomal RNA blast database from the NCBI `ftp://ftp.ncbi.nlm.nih.gov/blast/db/16S_ribosomal_RNA.tar.gz` Copy the url and retrieve the dataset with "wget" in the "data" directory. Extract the files with the "tar" command.

Download the SRR002326 dataset by going to `https://www.ebi.ac.uk/ena/data/view/SRS000608&display=html` via your **web browser**. You can find the files in the table of the "Read files" tab and copy the URL of the SRR002326 "File 1" file from the "Fastq files (ftp)" column. Retrieve the file

with "wget". The fastq file can be uncompressed with gunzip. Since you can pipe the output of one program to the next we will not uncompress the file entirely, but do this in a "streaming" way. With the -c option the content of the file will be displayed on screen. Try the following:

```
gunzip −c SRR002326.fastq.gz |head −n 12
```

Sequence entries in fastq format consist of 4 lines, see `https://en.wikipedia.org/wiki/FASTQ_format`. We can count the sequences by using a combination of "grep" and "wc". We will do this in two steps, so you can see what is happening.

```
gunzip −c SRR002326.fastq.gz | grep '^@' |head
gunzip −c SRR002326.fastq.gz | grep '^@' |wc −l
```

## References


[1] Carola Simon and Arnim Wiezer and Axel W. Strittmatter and Rolf Daniel, *Phylogenetic Diversity and Metabolic Potential Revealed in a Glacier Ice Metagenome*, Appl. Environ. Microbiol., 2009, 75(23), pp. 7519-7526.


# 5 Quality control

In the first step the quality of the sequences are examined with the program FastQC. `http://www.bioinformatics.babraham.ac.uk/projects/fastqc/` Copy the url of the *Win/Linux zip file* in the download section (from your web browser) and fetch the FastQC zip file with "wget" from the command-line. Unzip the file with "unzip" and read the INSTALL.txt to see what else needs to be done before you can use FastQC. Add the "fastqc" tool to the PATH variable.

Once FastQC is set up you can examine the help text of fastqc and how to use it from the commandline.

```
fastqc −−help
```

Note that getting the help page can be different for programs. Here are a few examples

```
theprogram −−help
theprogram −help
theprogram −h
man theprogram
theprogram # if nothing else gave you the help page
```

Start FastQC in interactive mode and analyse the fastq file that you have downloaded earlier. There is also the possibility to start the program from the commandline. The benefit of this is that you can use the program in an automated analysis pipeline or to run it on multiple fastq files at once. Run the same analysis, but now from the commandline. After this a new zip file has been created with the results. You can "unzip" the file and examine the results with the "more" or "less" commands.

# 6 Data conversion

BLAST accepts input files in FASTA format, however, the sequence reads are in FASTQ format. There are several data conversion tools, such as FASTX (`http://hannonlab.cshl.edu/fastx_toolkit/`). However, the data conversion is simple enough to perform from the commandline. We need the first and second row, and change the "@" into a ">" sign. Here is an example using awk (from `http://seqanswers.com/forums/showthread.php?t=6888`)

```
awk 'BEGIN{P=1}{if(P==1||P==2){gsub("@",">");print}; if(P==4){P
    =0};P++}' input.fastq > output.fasta
```

and a similar one-line program in the Perl language

```
perl -ne '$c++; if ($c==1){$_=~/\@(\S+)/; print ">$1\n"} if($c==2){
    print "$_"} if ($c==4){$c=0}' input.fastq > output.fasta
```

Set up a one-line program to convert the gzipped fastq file to fasta format. The answer is at the end of this document.

# 7 Sequence alignment

In this section we will align the sequence reads against the 16S Microbial database with BLAST. The complete sequence file takes a long time to align, therefore we start with a small subset. Use the "head" command to get the first 1000 lines from the fasta file and store this in the file "test.fasta" Hint: type "man head" to see how you invoke this command. Answer at the end of this document.

BLAST has many options, but for a default alignment we only need to specify the database, the query sequences, and the output file.

```
blastn -db databasename -query sequences.fasta -out testblast.txt
```

Perform a blast search with the test data against the microbial database.

Inspect the blast results. Blast didn't find any sequence back in the database, so these are either not 16S microbial sequences, or the sequences differ to much from the sequences in the database. You can do a quick check for a few sequences using the web-based nucleotide BLAST (`http://www.ncbi.nlm.nih.gov/blast/`)

Inspect the help of blastn

```
blastn -help
```

The default program that is executed is "megablast", which performs a faster search, but doesn't allow for too many mismatches. In this case we might be better off with the "blastn" program. Execute the blastn program again with "-task blastn" and inspect the results.

To process the blast output it is easier to use a tabular output format. Change the output format to "tabular", restrict the number of alignments per query to 1, and change the output fields to include the organism name (sscinames) of the database hit.

Execute the same on the complete dataset and store the output in "blast-all.txt". Add an "&" sign after the command to run the program in the background. While blast is running we can move on to the next step.

# 8 Create a summary of the BLAST result

Make a one-line program to count the organisms in the experiment. Get the following columns in the output: count and organism name. Sort the results based on the count.

You can use the following commands:

- cut
- uniq
- sort

Use the "man" command to find out out how each command works (or use your favourite search engine). The testblast.txt file can be used for development.

With this command you can check if blast is ready

```
ps −fe | grep blastn
ps −fe | grep studentNN
```

or alternatively, use the "top" program or "jobs".

```
jobs
top
```

Hit the "u" key and fill in your username to see all the processes. With "q" you can exit the program.

Bonus question (if blast is still running): How would you track the progress of the running blast program?

When everything works and blast is ready you can run the same one-line program on the larger dataset. Which bacterium is dominant in this dataset? What are the characteristics of this bacterium and did you expect this to find in a glacier?

# 9 Answers

## 9.1 Getting data from the web

```
wget ftp://ftp.ncbi.nlm.nih.gov/blast/db/16S_ribosomal_RNA.tar.gz
tar zxvf 16S_ribosomal_RNA.tar.gz
wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR002/SRR002326/SRR002326.fastq
    .gz
```

## 9.2 Quality control

```
wget http://www.bioinformatics.babraham.ac.uk/projects/fastqc/fastqc_v0
    .11.9.zip
unzip fastqc_v0.11.9.zip
```

In the INSTALL.txt file you can read that you need to make the "fastqc" file executable. This can be done with

```
chmod 755 fastqc
```

FastQC can be started in interactive mode by entering "fastqc" without arguments. When one or more fastq files are given as argument these files will be processed in commandline mode.

```
fastqc SRR002326.fastq.gz
```

## 9.3 Data conversion

```
gunzip -c SRR002326.fastq.gz | awk 'BEGIN{P=1}{if(P==1||P==2){gsub(
    "@",">");print}; if(P==4){P=0};P++}' > SRR002326.fasta
```

## 9.4 Sequence alignment

Get the first 1000 lines of the fasta file:

```
head -n 1000 SRR002326.fasta > test.fasta
```

```
blastn -db 16S_ribosomal_RNA -query test.fasta -out testblast.txt
```

Change the output format and include taxonomy names of the database hits

```
blastn -task blastn -db 16S_ribosomal_RNA -query test.fasta -out
    testblast.txt -num_alignments 1 -outfmt '6 qseqid sseqid pident length
    mismatch gapopen qstart qend sstart send evalue bitscore sscinames'
```

The program probably gives an error message that it can't look up the taxonomy name. Download and extract the taxdb from the url in the error message and try again.

On the complete dataset:

```
blastn -task blastn -db 16S_ribosomal_RNA -query SRR002326.fasta -out
    blast-all.txt -num_alignments 1 -outfmt '6 qseqid sseqid pident length
    mismatch gapopen qstart qend sstart send evalue bitscore sscinames' &
```

### 9.5 Create a summary of the BLAST result

```
cut −f 13 testblast.txt |sort|uniq −c|sort −n
```

With the "cut" command one or more fields (columns) can be extracted from a tabular file. The "-f number" argument specifies the column(s) to retrieve. "sort" sorts the results alphabetically, the "-n" arguments sorts numerically. "uniq" make a sorted list unique, the "-c" option adds a count for each unique entry.

How to check the progress of blast? One way is to count the entries in the fasta file with the "grep" and "wc" command, and compare that to the amount of entries in the blast output file.