Bash_course

Lokesh

November 4, 2018

Shortcuts

There are a few shortcuts that may be used in bash. One is the history commands, “Up arrow”" means go back in the history of commands. “Down arrow” means go forward in the history of commands.

“Ctrl-a” go to beginning of the line

“Ctrl-e” go to end of the line

“Ctrl-d” remove character at the cursor position (or exit a command or shell)

“Ctrl-k” remove everything from the cursor to the end of the line

“Ctrl-u” remove everything before the cursor

“Ctrl-l” clear the terminal

“Ctrl-c” exit a running program

Remote login

This is the means of connecting from your local terminal and access bigger and more powerful computer clusters.

```
ssh bagnoud@vlogin4.csb.univie.ac.at
```

Click yes for the first question (if any). This question will not appear again.

Enter the password given to you by the teacher.

Then you can check what kinds of files Alex has in his home folder:

```
ls
```

Make a directory with your name for your files:

```
mkdir YOUR_NAME
cd YOUR_NAME
```

Then create a file on the remote computer and copy it to your local computer.

```
echo "Hello world" > test.txt
# Creates the file test.txt with the content Hello world
cat test.txt # Check the content
exit # Exit the session, you are now back on your own computer
```

Copying files from the remote computer

An alternative to exit is just “Ctrl-d”. Copy the created file to your own computer:

```
scp bagnoud@vlogin4.csb.univie.ac.at:~/YOUR_NAME/test.txt . # Note the dot
# The dot means to this directory
cat test.txt
# cat shows the content of the file and this is just to check
# that everything worked
```

If you want to copy an entire directory just add a -r (recursive) as an argu- ment. First we again login to the remote computer:

```
ssh bagnoud@vlogin4.csb.univie.ac.at
cd YOUR_NAME
mkdir Test
cd Test
touch a b c # touch creates empty files
ls -l # Check if true
exit


cd # Go home, if not at home already
scp -r bagnoud@vlogin4.csb.univie.ac.at:~/YOUR_NAME/Test .
ls -l ~/Test
```

Compression & archiving

Log-in back into Alex’s CUBE account!

Like in Windows there are commands for compression and archiving. Com- pression and uncompression are often carried out with “gzip” and “gunzip”. Both commands acts on a single file at the time. The original file is deleted when you run the command. Copy “paxillus.fna.gz” (which contains Illumina data) from Alex’s home directory (DO NOT USE “mv”) into your working directory. Take a look at the file size before and after “gunzip”.

```
ls -l -h paxillus.fna.gz # -l is long listing and -h is human readable(!)
gunzip paxillus.fna.gz # uncompress
ls -l -h paxillus.fna # the gz extension disappears after uncompressing
less paxillus.fna # Take a look at the file, end with q
```

“head -n” takes the n fist lines of the input file and outputs them. “tail” works the same, but takes the last lines of the file. Test:

```
head -10 paxillus.fna
tail -10 paxillus.fna
```

Archiving is carried out with the “tar” command. We first create two files, then tar them into one file, unpack them again and then we do the same procedure but now combining with compression.

```
head -10000 paxillus.fna > first5000.fna
tail -10000 paxillus.fna > last5000.fna
tar -cvf 10000.fna.tar first5000.fna last5000.fna
```

As you notice, options can be combined:

```
tar -cvf # is the same as tar -c -v -f
ls -lh # is the same as ls -l -h
```

“-c” means create an archive, “-v” is verbose (print info to the screen), “-f” shows that the next argument is the name of the file holding the archive. The list has to end with “-f”. The archive should end with the file extension “.tar”. Look at the files in the archive (“-t” means list the files):

```
tar -tf 10000.fna.tar
# Let's investigate the file sizes:
ls -l first5000.fna last5000.fna 10000.fna.tar
```

Does it make sense? Now we combine “tar” with compression using “tar” (this is the equivalent to zip in Windows; zip and unzip is also found in Linux).

```
tar -cvzf 10000.fna.tgz first5000.fna last5000.fna
#We added the -z to show that we want the files compressed. Take a look at file size
s:
ls -l first5000.fna last5000.fna 10000.fna.tar 10000.fna.tgz
```

Instead of using the file extension “tgz”, “tar.gz” may be used. Note that “tar” doesn't remove the original files like “gzip” and “gunzip” do.

To get the files back we do:

```
tar -xvf 10000.fna.tar # For uncompressed data
tar -xvzf 10000.fna.tgz # For compressed data
```

-x stands for extract. Again, notice that when you use gzip or gunzip the original file is deleted. Not so with tar, the original files remain in the directory. With the two last commands we overwrote the existing files.

Text manipulation

Take a look at the file:

```
cat paxillus.fna # Too fast! "Ctrl-c"
less paxillus.fna # Better
```

Move forward by using space. Go to the end with “G”. Go to the top with “p”. By typing a slash “/” a string can be searched throughout the file. Try entering a slash and then the string “TGTCTATGGA”. Do you find a match? Go to the next match by typing “n”. Search for the read “HISEQ1:132:D0T25ACXX:3:1104:17112:70958” (note that the string “D0T” contains a zero and not the letter O). It will take some time because it is in the bottom of the file. Quit by typing “q”.

GREP

“grep” takes a pattern, in this case “>”, and looks for that pattern in each line and outputs only these lines. To look at all id lines:

```
cat paxillus.fna | grep '>' # We have to quote > since it would otherwise mean redire
ct output to file

#See how many sequences the file contains:

cat paxillus.fna | grep -c '>' # -c stands for count lines
#See how many lines the file contains:

cat paxillus.fna | wc -l # wc is word count and the -l means count lines instead of w
ords
```

How many unique sequences are found in the file (only works if the entire sequence is on one line only)?

```
cat paxillus.fna | grep -v '>' | sort | uniq | wc -l
# grep -v excludes all lines containing the pattern
# sort sorts the output alpha numerically.
# uniq takes out the unique lines from a sorted list
```

Look at sequences starting with “ATGAA”?

```
cat paxillus.fna | grep -v '>' | grep "^ATGAA" # ^ means beginning of line
```

How many sequences starts with “ATGAA”?

```
cat paxillus.fna | grep -v '>' | grep -c "^ATGAA" # -c means count
cat paxillus.fna | grep -v '>' | grep "^ATGAA" | wc -l # same
```

How many sequences ends with “CAAGC”?

```
cat paxillus.fna | grep -v '>' | grep -c "CAAGC$"
# In this context $ means end of line (or string)
```

Calculate total read length:

```
cat paxillus.fna | grep -v '>' | tr -d "\n" | wc -c
# -d means delete and "\n" means newline. tr takes a list of characters as input
```

Find out if there are any non-standard nucleotides and count the sum:

```
cat paxillus.fna | grep -v '>' | tr -d "\n"acgtACGT | wc -c # None!
```

Extract the id and sequence for read “HISEQ1:132:D0T25ACXX:3:1104:17112:70958”:

```
cat paxillus.fna | grep -A 1 HISEQ1:132:D0T25ACXX:3:1104:17112:70958
# -A 1 means include the matching line and one line after.
# -B 3 would mean include the matching line and three lines before
# remember that this will only work if the entire sequence is in one line!
```

Make the sequence reverse compliment.

```
cat paxillus.fna | grep -A 1 HISEQ1:132:D0T25ACXX:3:1104:17112:70958 | grep -v '>' |
 rev | tr acgtACGT tgcaTGCA
```

CUT

making tab delimited files or csv files:

```
cat paxillus.fna | grep '>' | head -10 | cut -d ':' -f 5-7 | tr : "\t"
# -d means delimiter, -f the field (column) number, "\t" is tab
# here if you use "," instead of "\t", then the resulting file would be a csv file.
```

cut “cuts” out columns from a file. Take out position 50-100 from the sequence of HISEQ1:132:D0T25ACXX:3:1103:19197:80091

```
cat paxillus.fna | grep -A 1 HISEQ1:132:D0T25ACXX:3:1103:19197:80091 | grep -v '>' |
 cut -b 50-100
# here cut -b means take out character 50 to 100
```

SED

“sed” is used mainly for replaceing characters and strings!

Replace the string HISEQ1 with the name of the organism:

```
cat paxillus.fna | sed s/HISEQ1/Paxillus/
```

“sed s///” substitutes a string (here HISEQ1) with another string (here Paxillus). It only replaces the first occurrence on each line: To change every occurrence a so called modifier is used:

```
cat paxillus.fna | sed s/:/@/ # Replaces first occurrence
cat paxillus.fna | sed s/:/@/g # Replaces all occurrences, g stands for global
cat paxillus.fna | tr : @ # A better alternative
```

Save the new output to file:

```
cat paxillus.fna | sed s/HISEQ1/Paxillus/ > newPaxillus.fna
```

Make all sequences to RNA sequences:

```
cat newPaxillus.fna | tr tT uU
```