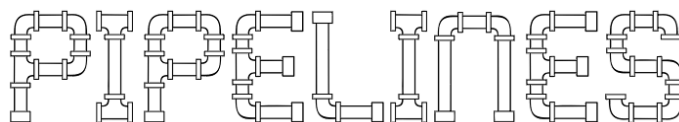


NGS Introduction Course

Practical two, Pipelines

Jeroen Laros, Michiel van Galen

Friday, 4 April 2014



1 A simple analysis pipeline

Log in to the virtual machine as described in the first practical. Finish these exercises first then continue with this one.

Analysis pipelines can vary from a couple of very simple lines of code, to complex frameworks working on multiple computing nodes. However, the idea behind it is always the same. Make your analysis easier, while at the same time it becomes reproducible, documented and easier to share.

Today we will use bash to create a basic pipeline. Unknowingly, you have already worked with bash. Everything you've typed into the terminal so far is basically part of bash. To create a pipeline, all we need to do is create a file with the “.sh” extension and simply sum up the steps you want to include.

First, let's see how to make your script executable. We need to change the permissions. This way you can simply run the file directly without having to specify the interpreter. In Linux we use chmod for this.

```
$ chmod +x file.sh
```

One last additional piece of information your system is missing, is which interpreter it should use. We store this in the very first line of the script and is written like this “#!/bin/bash”.

This is all you need to know to write your first pipeline. Bash offers a lot more possibilities than just using Linux power tools, navigation and starting other software. For example, to make the pipeline easier to re-use with other data, we can give a bash script parameters. Just like Bowtie, where you can supply your fastq file. This is done by using the reserved variable \$1 inside the script. Everywhere in your script this variable will be replaced with the first argument you give on the command line.

The next exercise will give you an idea on how to structure a simple bash pipeline. Complete the exercise and try to understand what is going on in the script:

- Write the lines you see below to a file “Hello.sh”
- Make the file executable “chmod +x Hello.sh”
- Run it with an argument “./Hello.sh yourname” and see what it does

- Comments start with `#` and are used to document your work

```
#!/bin/bash
echo Hello \"$1\", the date is:
# The next line will print the date.
date
```

With this knowledge you should be able to write your own short bash pipeline. Combine this with what you have learned in the first practical and try to write a script that combines the tools from the first practical into a pipeline:

- Include these steps:
 - Align a fastq file to the mtDNA
 - Convert the SAM to a sorted BAM
 - Call variants
 - Annotate the variants
- Keep these things in mind when writing the script:
 - The fastq file should be the first argument
 - Use comments to explain what happens
 - Make it executable
 - Bonus: Can you think of adding more arguments? Maybe for annotation?

Hints and summary:

- A comment line starts with an `#`
- `$1` is the first argument, `$2` the second, etc...
- Make your pipeline executable with “`chmod +x file`”

Now you know how to write a basic pipeline. In bash you can also work with conditions, loops and error handling but this is beyond the scope of this course. If you like to know more there is plenty to read on the web or visit one of our other courses. This is the end of the practical.

Thank you for participating!