Implementing Scalable Bioinformatic Workflows in Snakemake and Nextflow

Nathan S. Watson-Haigh
Radosław Suchecki

Across Australia

Aug/Sep 2019

Licensing

Contents

Licensing 3

Contents 4

Workshop Information 5
- The Trainers . . . 6
- Providing Feedback . . . 7
- Document Structure . . . 7

Introduction to Snakemake 9
- Key Learning Outcomes . . . 10
- Resources Required . . . 10
- Useful Links . . . 10
- Setting Up Your Environment . . . 11

Data Quality 17
- Key Learning Outcomes . . . 18
- Resources You'll be Using . . . 18
- Useful Links . . . 18
- Introduction . . . 19

Space for Personal Notes or Feedback 23

Workshop Information

The Trainers

Dr. Nathan S. Watson-Haigh
Senior Bioinformatician
Bioinformatics Hub, University of Adelaide
nathan.watson-haigh@adelaide.edu.au

Dr. Radosław Suchecki
Research Scientist
Crop Bioinformatics and Data Science, CSIRO
rad.suchecki@csiro.au

Providing Feedback

While we endeavour to deliver a workshop with quality content and documentation in a venue conducive to an exciting, well run hands-on workshop with a bunch of knowledgeable and likable trainers, we know there are things we could do better.

Whilst we want to know what didn't quite hit the mark for you, what would be most helpful and least depressing, would be for you to provide ways to improve the workshop. i.e. constructive feedback. After all, if we knew something wasn't going to work, we wouldn't have done it or put it into the workshop in the first place!

Clearly, we also want to know what we did well! This gives us that "feel good" factor which will see us through those long days and nights in the lead up to such hands-on workshops!

With that in mind, we'll provide a some high tech mechanism through which you can provide anonymous feedback during the workshop:

1. Some empty ruled pages at the back of this handout. Use them for your own personal notes or for writing specific comments/feedback about the workshop as it progresses.

Document Structure

We have provided you with an electronic copy of the workshop's hands-on tutorial documents. We have done this for two reasons: 1) you will have something to take away with you at the end of the workshop, and 2) you can save time (mis)typing commands on the command line by using copy-and-paste.

We advise you to use Acrobat Reader to view the PDF. This is because it properly supports some features we have implemented to ensure that copy-and-paste of commands works as expected. This includes the appropriate copy-and-paste of special characters like tilde and hyphens as well as skipping line numbers for easy copy-and-past of whole code blocks.

> While you could fly through the hands-on sessions doing copy-and-paste, you will learn more if you use the time saved from not having to type all those commands, to understand what each command is doing!

The commands to enter at a terminal look something like this:

```
tophat --solexa-quals -g 2 --library-type fr-unstranded -j \
    annotation/Danio_rerio.Zv9.66.spliceSites -o tophat/ZV9_2cells \
    genome/ZV9 data/2cells_1.fastq data/2cells_2.fastq
```

The following styled code is not to be entered at a terminal, it is simply to show you the syntax of the command. You must use your own judgement to substitute in the correct arguments, options, filenames etc

```
tophat [options]* <index_base> <reads_1> <reads_2>
```

The following is an example of how R commands are styled:

```
R --no-save
library(plotrix)
data <- read.table("run_25/stats.txt", header=TRUE)
weighted.hist(data$short1_cov+data$short2_cov, data$lgth, breaks=0:70)
q()
```

The following icons are used in the margin, throughout the documentation to help you navigate around the document more easily:

- ℹ Important
- 🗎 For reference
- 👣 Follow these steps
- Q Questions to answer
- ! Warning - STOP and read
- ✸ Bonus exercise for fast learners
- ✸ Advanced exercise for super-fast learners

Introduction to Snakemake

Primary Author(s):
Nathan S. Watson-Haigh nathan.watson-haigh@adelaide.edu.au

Contributor(s):

Key Learning Outcomes

After completing this module the trainee should be able to:

- Install Snakemake in a conda environment
- Execute a Snakemake workflow
- Use the provided ”profile” to execute jobs on a compute cluster
- Write simple Snakemake rules capable of generating some output(s) by executing some code which perates on some input(s)

Resources Required

For the purpose of this training you need access to:

- A compute cluster with the `module` command available to you for loading software
- https://sylabs.io/singularity/Singularity - available as a module on the above cluster
- https://www.anaconda.com/distribution/conda - available as a module on the above cluster

Tools Used

Snakemake
https://snakemake.readthedocs.io

Graphviz
https://www.graphviz.org

Useful Links

Slurm Documentation
https://slurm.schedmd.com/documentation.html

Setting Up Your Environment

For the purpose of the workshop we will be working on the head node of an HPC cluster running https://slurm.schedmd.com/documentation.htmlSlurm. This is the most likely infrastructure that fellow bioinformaticians already find themselves using on a regular basis. We also assume that the cluster provides the `module` command for you to load software and the modules `Anaconda3` and `Singularity` are available to use.

The execution of the Snakemake workflow will actually take place on the cluster head node with jobs being submitted to Slurm for queing and processing. From the head node, Snakemake will monitor the submitted jobs for their completion status and submit new jobs as dependent jobs complete sucessfully.

Connect to the Cluster Head Node

First up, lets connect to the head node of the HPC cluster using `ssh`.

See your local facilitator for connection details. You should have one user account per person.

Install Snakemake

The recommended installation route for Snakemake is through a conda environemnt (https://snakemake.readthedocs.io/en/stable/getting_started/installation.html). As such, you need Anaconda3, usually avaiable to you on your cluster via the module system.

```
# Load the Anaconda3 module on your cluster
# If it's unavailable contact the cluster sysadmin
module load \
  Anaconda3

# Create a new conda environment for snakemake
# We use a specific version for reproducibility reasons
# https://anaconda.org/search?q=snakemake
SNAKEMAKE_VERSION="5.5.4"

conda create \
  --name snakemake \
  --channel bioconda --channel conda-forge \
  --yes \
  snakemake="${SNAKEMAKE_VERSION}"
```

That's all there is to installing Snakemake.

To activate the environment and make `snakemake` available on the command line simply:

```
# activate the conda environment you created above
source activate snakemake
```

While waiting for others to catch up, why not have a look into how you would go about updating `Snakemake` within this conda environment if there is a new version available.

Running a Workflow

We are going to work with an example Snakemake workflow to demonstrate how they are run before we get stuck into writing our own. This example workflow consists of the following steps:

- Running FastQC across the raw reads
- Aggregating the raw read FastQC reports using MultiQC
- Performing adapter, quality, and read length filtering using Trimmomatic
- Running FastQC across the QC'd reads
- Aggregating the QC read FastQC reports using MultiQC
- Index the reference FASTA file
- Perform a `bwa-mem` read alignment

Lets get the example workflow:

```
# Clone the repository
git clone \
    https://github.com/UofABioinformaticsHub/2019_EMBL-ABR_Snakemake_webinar \
    ./example_workflow

cd example_workflow/snakemake-tutorial
```

The most basic of Snakemake commands will execute the workflow, running jobs on the head node (your cluster sysadmin will probably growl at you). Lets be nice, and just do a `dryrun` for now:

```
snakemake \
  --dryrun
```

Using the Snakemake help, how do you get Snakemake to print the shell commands it will execute for each job?

This is an important note to the reader. The paragraph get's it own margin icon and formatting to ensure it stands out.

There are 10 characters which have special meaning in LaTeXand to have them displayed as literal characters we need to do something special. The first seven can simply be prepended by a backslash; for the other three, we must use the macros `\textasciitilde`, `\textasciicircum`, `\textbackslash`.

& % $ # _ { }

~

^

\

Highly redundant coverage (>15X) of the genome can be used to correct sequencing errors in the reads before assembly and errors. Various k-mer based error correction methods exist but are beyond the scope of this tutorial.

In order to use a single character to encode Phred qualities, ASCII characters are used (`http://shop.alterlinks.com/ascii-table/ascii-table-us.php`). All ASCII characters have a decimal number associated with them but the first 32 characters are non-printable (e.g.

Because ASCII characters < 33 are non-printable, using the Phred+33 encoding was not possible. Therefore, they simply moved the offset from 33 to 64 thus

schema it is not always possible to identify what encoding is in use. For example, if the only characters seen in the quality string are (`@ABCDEFGHI`), then it is impossible to know if you have really good Phred+33 encoded qualities or really bad Phred+64 encoded qualities.

Tables

Writing tables in LaTeXis HARD! Rather than write them from scratch, I'd head over to http://www.tablesgenerator.com/latex_tables, generate the table you want using the interactive page and copy the resulting LaTeXinto your source file.

Item		
Animal	Description	Price ($)
Gnat	per gram	13.65
	each	0.01
Gnu	stuffed	92.50
Emu	stuffed	33.33
Armadillo	frozen	8.99

Figures

No handout is complete without figures, screenshots or similar.

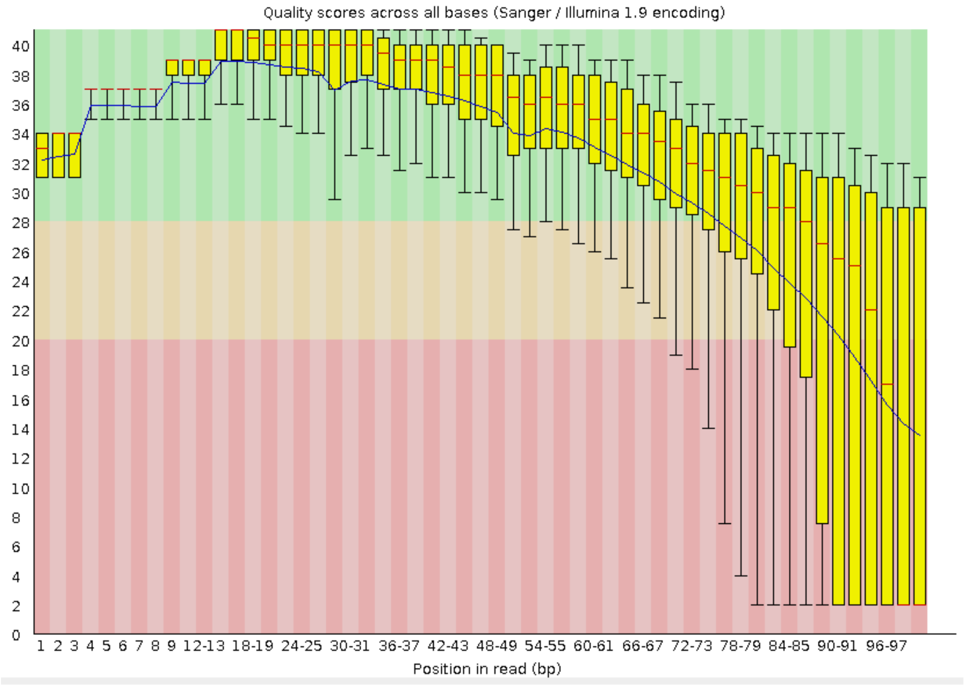

Figure 1: Per base sequence quality plot for `bad_example.fastq`.

Maths Environment

I won't pretend to be an expert in LaTeXbut I do know that one of the strengths of LaTeXis it's ability to allow mathematical formulas to be constructed and typeset very easily. Firstly, you have to tell LaTeXwhat text is going to contain mathematical symbols.

For inline display, simply surround the relevant text by dollar signs: `$Q(A) =-10 log10(P(\sim A))$`

For more complex mathematical formulas which need to be displayed separately, use the `\begin{displaymath} ... \end{displaymath}` environment to wrap around your formulas.

$$Q(A) = -10log10(P(\sim A))$$

Questions and Answers

It is useful to maintain the answers to questions together in the same LaTeXsource. By using the `\begin{answer} ... \end{answer}` environment we can have the enclosed answers excluded from the trainee's version of the handout while including it in the trainer's version of the handout.

> We can ask a simply question to test if the trainee is understanding what they are doing.

Code For Trainees to Copy/Type

Different people have different opinions on whether it is a good idea to provide the commands for trainees to copy-and-paste. In our experience, there is a huge amount of time wasted by novices typing commands incorrectly or changing filenames which affects commands you might run later on. We also think that it's a good idea to pose regular questions or ask the trainees to modify a previous command. This way you can catch out those whoe are just trying to get ahead by blindly copying-and-pasting.

To provide a nicely formatted, copy-and-pastable command, simply wrap it in the `\begin{lstlisting} ... \end{lstlisting}` environment. Long commands will be wrapped automatically and bash line continuation characters (\) inserted where required.

```
cd ~/QC
fastx_clipper -h
fastx_clipper -v -Q 33 -l 20 -M 15 -a GATCGGAAGAGCGGTTCAGCAGGAATGCCGAG \
    -i bad_example.fastq -o bad_example_clipped.fastq
```

Data Quality

Primary Author(s):
Jane Bloggs jane.bloggs@example.com
Joe Bloggs (joe.bloggs@example.com)

Contributor(s):
John Doe john.dow@example.com

Key Learning Outcomes

After completing this module the trainee should be able to:

- Assess the overall quality of NGS sequence reads
- Visualise the quality, and other associated matrices, of reads to decide on filters and cutoffs for cleaning up data ready for downstream analysis
- Clean up and pre-process the sequences data for further analysis

Resources You'll be Using

Tools Used

FastQC
`http://www.bioinformatics.babraham.ac.uk/projects/fastqc/`

FASTX-Toolkit
`http://hannonlab.cshl.edu/fastx_toolkit/`

Picard
`http://picard.sourceforge.net/`

Useful Links

FASTQ Encoding
`http://en.wikipedia.org/wiki/FASTQ_format#Encoding`

Introduction

This is an important note to the reader. The paragraph get's it own margin icon and formatting to ensure it stands out.

There are 10 characters which have special meaning in LaTeXand to have them displayed as literal characters we need to do something special. The first seven can simply be prepended by a backslash; for the other three, we must use the macros `\textasciitilde`, `\textasciicircum`, `\textbackslash`.

& % $ # _ { }

~

^

\

Highly redundant coverage (>15X) of the genome can be used to correct sequencing errors in the reads before assembly and errors. Various k-mer based error correction methods exist but are beyond the scope of this tutorial.

In order to use a single character to encode Phred qualities, ASCII characters are used (http://shop.alterlinks.com/ascii-table/ascii-table-us.php). All ASCII characters have a decimal number associated with them but the first 32 characters are non-printable (e.g.

Because ASCII characters $<$ 33 are non-printable, using the Phred+33 encoding was not possible. Therefore, they simply moved the offset from 33 to 64 thus

schema it is not always possible to identify what encoding is in use. For example, if the only characters seen in the quality string are (`@ABCDEFGHI`), then it is impossible to know if you have really good Phred+33 encoded qualities or really bad Phred+64 encoded qualities.

Tables

Writing tables in LaTeXis HARD! Rather than write them from scratch, I'd head over to http://www.tablesgenerator.com/latex_tables, generate the table you want using the interactive page and copy the resulting LaTeXinto your source file.

Item		
Animal	Description	Price ($)
Gnat	per gram	13.65
	each	0.01
Gnu	stuffed	92.50
Emu	stuffed	33.33
Armadillo	frozen	8.99

Figures

No handout is complete without figures, screenshots or similar.

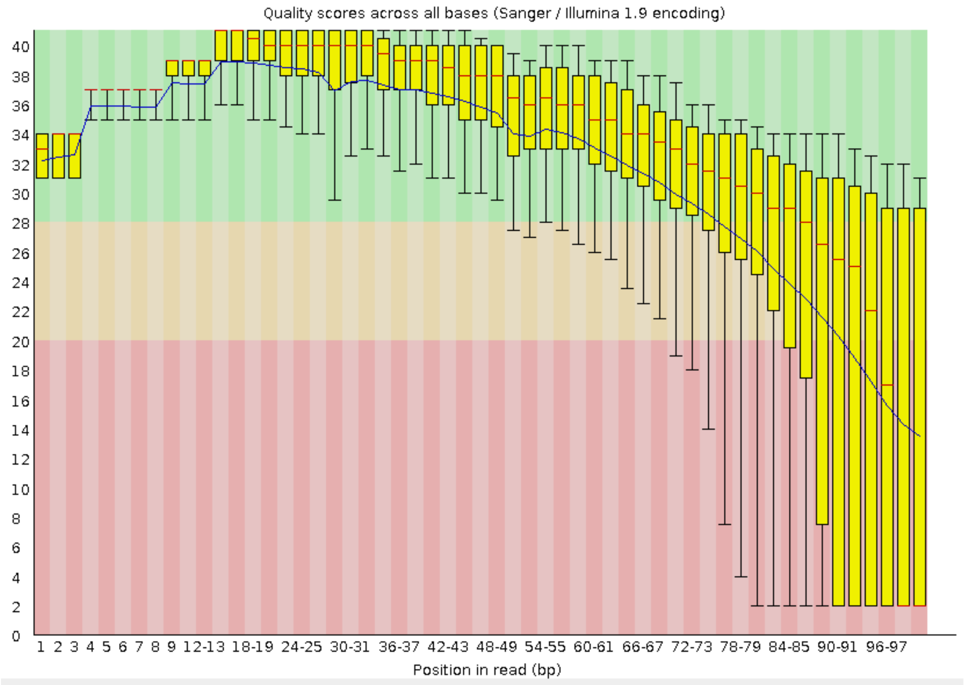

Figure 2: Per base sequence quality plot for `bad_example.fastq`.

Maths Environment

I won't pretend to be an expert in LaTeXbut I do know that one of the strengths of LaTeXis it's ability to allow mathematical formulas to be constructed and typeset very easily. Firstly, you have to tell LaTeXwhat text is going to contain mathematical symbols.

For inline display, simply surround the relevant text by dollar signs: `$Q(A) =-10 log10(P(\sim A))$`

For more complex mathematical formulas which need to be displayed separately, use the `\begin{displaymath} ... \end{displaymath}` environment to wrap around your formulas.

$$Q(A) = -10log10(P(\sim A))$$

Questions and Answers

It is useful to maintain the answers to questions together in the same LaTeXsource. By using the `\begin{answer} ... \end{answer}` environment we can have the enclosed answers excluded from the trainee's version of the handout while including it in the trainer's version of the handout.

> We can ask a simply question to test if the trainee is understanding what they are doing.

Code For Trainees to Copy/Type

Different people have different opinions on whether it is a good idea to provide the commands for trainees to copy-and-paste. In our experience, there is a huge amount of time wasted by novices typing commands incorrectly or changing filenames which affects commands you might run later on. We also think that it's a good idea to pose regular questions or ask the trainees to modify a previous command. This way you can catch out those whoe are just trying to get ahead by blindly copying-and-pasting.

To provide a nicely formatted, copy-and-pastable command, simply wrap it in the `\begin{lstlisting} ... \end{lstlisting}` environment. Long commands will be wrapped automatically and bash line continuation characters (\\) inserted where required.

```
cd ~/QC
fastx_clipper -h
fastx_clipper -v -Q 33 -l 20 -M 15 -a GATCGGAAGAGCGGTTCAGCAGGAATGCCGAG \
   -i bad_example.fastq -o bad_example_clipped.fastq
```

Space for Personal Notes or Feedback