

Reproducible bioinformatics

from a user's perspective

Tom Harrop

The University of Otago

tom.harrop@otago.ac.nz

[@tharrop_](#)

2020-02-12

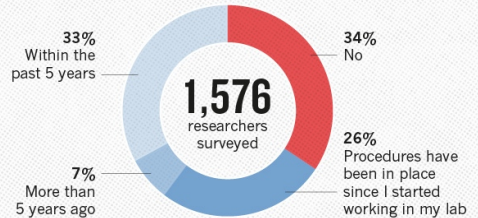
IS THERE A REPRODUCIBILITY CRISIS?



©nature

HAVE YOU ESTABLISHED PROCEDURES FOR REPRODUCIBILITY?

Among the most popular strategies was having different lab members redo experiments.



©nature

What is reproducibility?

Reproduce: under identical conditions to the previous result, repeat the analysis and get the **exact** same result

In bioinformatics:

- **same data**
- **same methodology** (code)
- **same result**

Guidelines for reproducible analysis:

1. Don't modify raw data
2. Record the code
3. Capture the computing environment

Interactive analysis may be hard to reproduce

Examples:

- install software locally
- use software installed by the admin
- paste commands from a text file into the console
- save a set of scripts to run in order

Possible issues:

- will it run again?
- are all the steps documented?
- is the recorded code exactly what was run?
- are the steps in the right order?

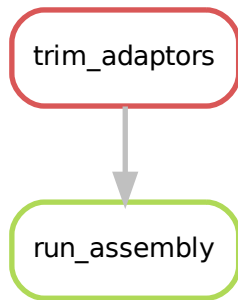
Workflow managers force you to record every step

Define steps in `my_workflow.txt`

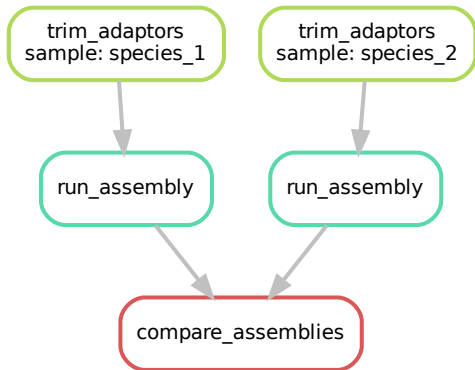
```
step trim_adaptors:  
  input:  'data/raw_reads/{sample}.fastq'  
  output: 'output/trimmed/{sample}.fastq'  
  shell:  'trim_adaptors --raw_reads={input} > {output}'  
  
step run_assembly:  
  input:  'output/trimmed/{sample}.fastq'  
  output: 'output/assemblies/{sample}.fasta'  
  shell:  'choice_assembler --reads={input} > {output}'
```

Run:

```
workflow_manager my_workflow.txt run_assembly
```



Reproducibility and convenience



- The code *is* the documentation
- Scale the same code to different data
- Version control → versioned results

Lots of good options:

snakemake : python3

nextflow : java

CWL : 'vendor-neutral specification'

drake : R

make : DIY

Reproducible computing environment

Software has

- a **version**,
- other software **dependencies** (with versions)
- all with **system dependencies**

e.g. DESeq2

DESeq2_1.26.0

Bioconductor 3.10.1

libblas3 3.8.0, libc6 2.30, *etc.*

Software containers

- Isolated, complete environment (a mini OS)
- Contain specific version of software with dependencies
- Mobility of compute
- Reproducibility
- Singularity can run on traditional HPC



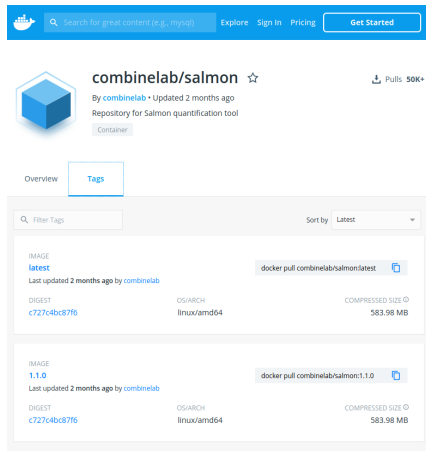
docker



Getting software in containers

- Some developers provide docker containers

```
singularity pull \  
  --name salmon_1.1.0.sif \  
  docker://combinelab/salmon:1.1.0
```



The screenshot shows the Docker Hub interface for the `combinelab/salmon` repository. The repository is a container image created by `combinelab`, updated 2 months ago, and used as a repository for the Salmon quantification tool. It has over 50K pulls. The 'Tags' tab is selected, showing a list of image tags. The 'latest' tag is the most recent, last updated 2 months ago. Below it, the '1.1.0' tag is listed. Both tags have the same digest (`c727c4bc87f6`) and OS/ARCH (`linux/amd64`), with a compressed size of 583.98 MB. The interface includes a search bar, navigation links (Overview, Tags), and a 'docker pull' command for each tag.

IMAGE	OS/ARCH	COMPRESSED SIZE
latest Last updated 2 months ago by combinelab	linux/amd64	583.98 MB
1.1.0 Last updated 2 months ago by combinelab	linux/amd64	583.98 MB

Getting software into containers

- Often have to build our own containers

Singularity.bwa_0.7.17

Bootstrap: docker

From: ubuntu:18.10

%labels

VERSION "BWA 0.7.17"

%post

apt-get update

apt-get install -y bwa

%runscript

exec /usr/bin/bwa "\$@"

The screenshot shows the Singularity Hub web interface. At the top, there's a navigation bar with 'SINGULARITYHUB' and links for 'Collections', 'About', 'User Guide', 'Get Help', and a search icon. A user profile 'TomHarrop' is visible in the top right. The main content area displays the container 'TomHarrop/singularity-containers:bwa_0.7.17' in a red-bordered box. Below this, there are buttons for 'FREEZE' and 'Q'. A red 'Build Spec' button is prominent. To the right of the 'Build Spec' button are links for 'Labels' and 'Log'. The 'Build Spec' section is expanded, showing a list of build steps: 1. Bootstrap: docker, 2. From: ubuntu:18.10, 3. (empty), 4. %help, 5. Container for BWA 0.7.17, 6. http://bio-bwa.sourceforge.net/, 7. (empty), 8. %labels, 9. (empty), 10. VERSION "BWA 0.7.17", 11. (empty), 12. (empty), 13. %post, 14. (empty), 15. # install dependencies via apt, 16. apt update, 17. apt install -y \, 18. bwa, 19. (empty), 20. %runscript, 21. (empty), 22. exec /usr/bin/bwa "\$@".

On the right side, there's a 'Description' section with 'None'. Below it is the 'Build Metrics' section, which includes: Size (MB): 101, Tag: bwa_0.7.17, Commit: 3c62cd582cafcac61867efd8361bb413d67e6f37 (highlighted in a red box), Branch: master, Version (file hash): eb49586488ed8fa5ba71da34f2fabc1a (highlighted in a red box), Build Time: 40 seconds, Version 2.5.0-feature-squashbuild-secbuild-2.5.0.gdd162fb, and Version 2.5.

Workflow managers support containers

```
step trim_adaptors:
  input:          'data/raw_reads/{sample}.fastq'
  output:         'output/trimmed/{sample}.fastq'
  singularity:    'docker://my_repos/trim_adaptors:2.9'
  shell:          'trim_adaptors --raw_reads={input} > {output}'

step run_assembly:
  input:          'output/trimmed/{sample}.fastq'
  output:         'output/assemblies/{sample}.fasta'
  singularity:    'shub://my_repos/choice_assembler:1.5'
  shell:          'choice_assembler --reads={input} > {output}'
```

Some barriers to container usage

- **Building containers can be painful** if the dependencies are disorganised
- **Duplication of effort**
- **Some software shouldn't go in a container** because of “unfortunate licensing issues”
 - DTU software *e.g.* rnammer, tmhmm
 - GATech: GeneMark
 - GIRInst's RepBase
- **Getting Singularity installed**

Reproducible analysis stack

Guidelines:

1. Don't modify raw data
2. Record the code
(with version control)
3. Capture the computing
environment

Stack:

```
md5sum raw_reads.fastq? chmod 444?  
+ Workflow manager (snakemake, nextflow)  
+ VCS (git)  
+ Software containers (Singularity)
```

Getting started

Reproducibility for bioinformatics:

- Joep de Ligt: *Scalable workflows and reproducible data analysis for genomics*
- plenty of online talks e.g. [Adam Labadorf](#) of Boston Uni

Workflow managers:

- [Snakemake Tutorial](#)
- Nextflow: [Get started](#)

Software containers:

- Blair Bethwaite: *Containers in HPC* Tutorial
- Singularity [Quick Start](#)

Version control:

- memorise a handful of `git` commands