Reproducible bioinformatics

from a user's perspective

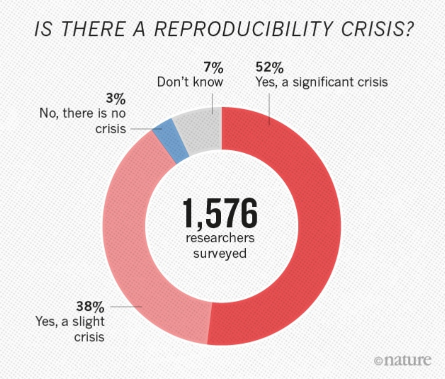
Tom Harrop

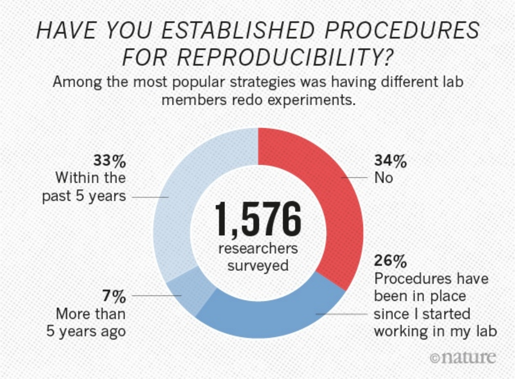
The University of Otago

tom.harrop@otago.ac.nz

@tharrop_

2020-02-12





Jenny Bryan https://youtu.be/7oyiPBjLAWY & Nature News, 10.1038/533452a

What is reproducibility?

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

In bionformatics:

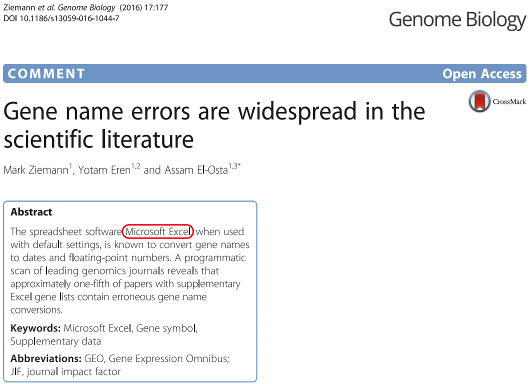
- **same data**
- **same methodology** (code)
- **same result**

For reproducible bioinformatics:

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

Adapted from Adam Labadorf’s slides

No peeking at the data



Ziemann *et al. Genome Biology* (2016) 17:177
DOI 10.1186/s13059-016-1044-7

Genome Biology

COMMENT **Open Access**

CrossMark

Gene name errors are widespread in the scientific literature

Mark Ziemann[1], Yotam Eren[1,2] and Assam El-Osta[1,3*]

Abstract

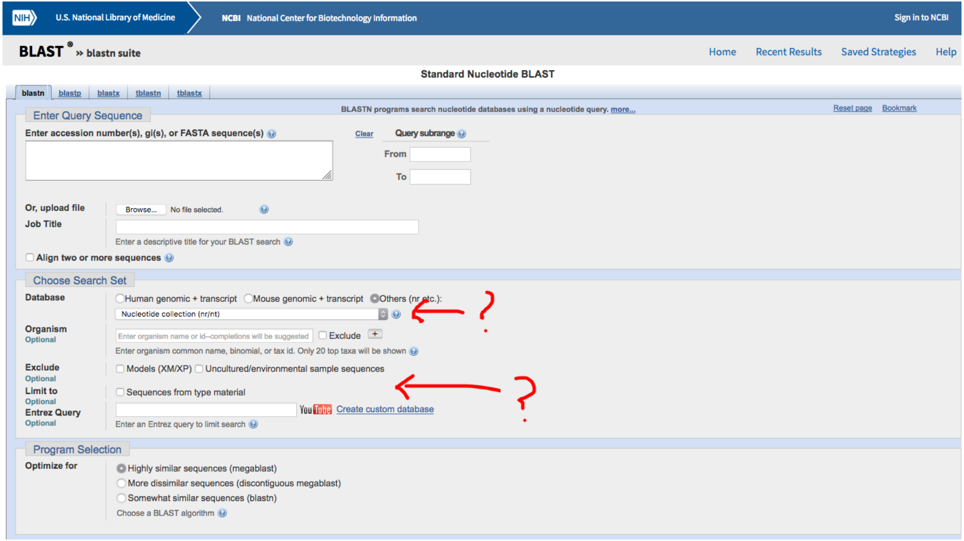
The spreadsheet software Microsoft Excel, when used with default settings, is known to convert gene names to dates and floating-point numbers. A programmatic scan of leading genomics journals reveals that approximately one-fifth of papers with supplementary Excel gene lists contain erroneous gene name conversions.

Keywords: Microsoft Excel, Gene symbol, Supplementary data

Abbreviations: GEO, Gene Expression Omnibus; JIF, journal impact factor

Ziemann *et al*., 2016. 10.1186/s13059-016-1044-7

Point-and-click software is less likely to be reproducible



U.S. National Library of Medicine | NCBI National Center for Biotechnology Information | Sign in to NCBI

BLAST® » blastn suite | Home | Recent Results | Saved Strategies | Help

Standard Nucleotide BLAST

blastn | blastp | blastx | tblastn | tblastx

BLASTN programs search nucleotide databases using a nucleotide query. more... | Reset page | Bookmark

Enter Query Sequence

Enter accession number(s), gi(s), or FASTA sequence(s) | Clear | Query subrange

From

To

Or, upload file: Browse... No file selected.

Job Title:

Enter a descriptive title for your BLAST search

□ Align two or more sequences

Choose Search Set

Database: ○ Human genomic + transcript ○ Mouse genomic + transcript ● Others (nr etc.):

Nucleotide collection (nr/nt) ← ?

Organism (Optional): Enter organism name or id--completions will be suggested □ Exclude +

Enter organism common name, binomial, or tax id. Only 20 top taxa will be shown

Exclude (Optional): □ Models (XM/XP) □ Uncultured/environmental sample sequences

Limit to (Optional): □ Sequences from type material ← ?

Entrez Query (Optional): YouTube Create custom database

Enter an Entrez query to limit search

Program Selection

Optimize for:

● Highly similar sequences (megablast)

○ More dissimilar sequences (discontiguous megablast)

○ Somewhat similar sequences (blastn)

Choose a BLAST algorithm

Running on the fly probably won’t be reproducible

Examples:

- install software locally
- use software installed by the admin
- type your commands directly into the console and hit enter!
- save a set of scripts to run in order

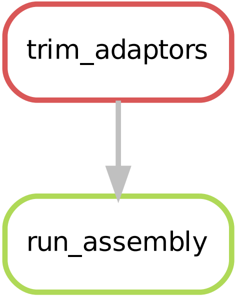
Possible issues:

- will it run again?
- are **all** the steps documented?
- is the code you recorded the same as the code you ran?
- did you correctly record the order of steps?

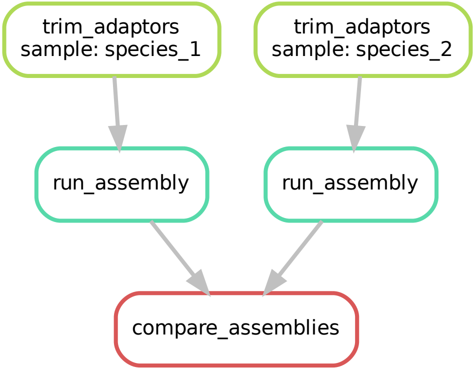
Workflow managers force you to record every step

```
rule trim_adaptors:
    input:  'data/raw_reads/{sample}.fastq',
    output: 'output/trimmed/{sample}.fastq'
    shell:  'trim_adaptors --raw_reads={input} > {output}'

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



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*.

Adapted from Adam Labadorf's slides

Reproducible computing environment

On our department's hardware:

```
salmon --version
```

```
salmon 0.9.1
```

***e.g.* Ubuntu 19.10**:

```
apt policy salmon
```

```
salmon:
  Installed: (none)
  Candidate: 0.12.0+ds1-1
  Version table:
     0.12.0+ds1-1 500
        500 http://nz.archive.ubuntu.com/ubuntu eoan/universe amd64 Packages
```

Software containers

- Isolated, complete environment (a mini OS)
- Contain specific version of software with dependencies

Singularity:

- Mobility of compute
- Reproducibility
- Support on existing traditional HPC

Singularity containers

Running directly:

```
salmon --help
```

```
Error in running command bash
```

Running with Singularity:

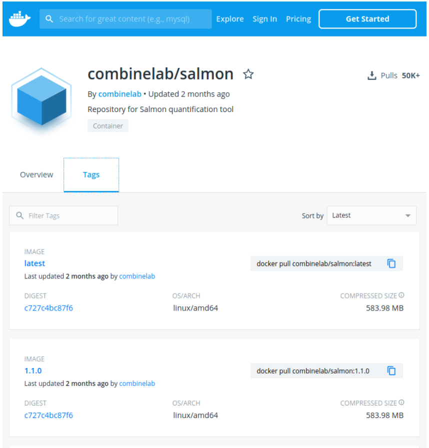
```
singularity exec \
  salmon_1.1.0.sif \
  salmon --help
```

```
Usage:  salmon -h|--help or
        salmon -v|--version or
        salmon -c|--cite or
        salmon [--no-version-check] <COMMAND> [-h | options]
```

Getting software in containers

- Some developers provide docker containers

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



combinelab/salmon

Pulls 50K+

By combinelab • Updated 2 months ago

Repository for Salmon quantification tool

Container

IMAGE
latest
Last updated **2 months ago** by combinelab

```
docker pull combinelab/salmon:latest
```

DIGEST c727c4bc87f6 | OS/ARCH linux/amd64 | COMPRESSED SIZE 583.98 MB

IMAGE
1.1.0
Last updated **2 months ago** by combinelab

```
docker pull combinelab/salmon:1.1.0
```

DIGEST c727c4bc87f6 | OS/ARCH linux/amd64 | COMPRESSED SIZE 583.98 MB

hub.docker.com

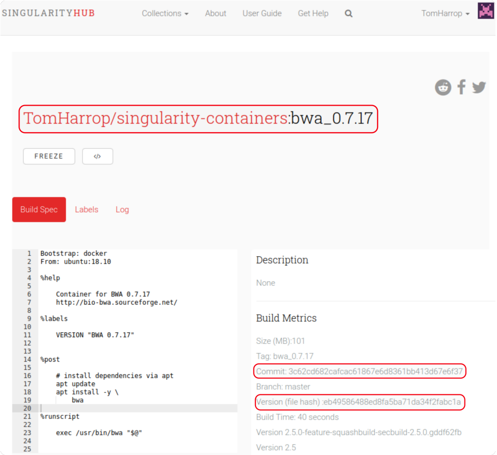
Getting software into containers

- Usually have to build it yourself

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 "$@"
```



TomHarrop/singularity-containers:bwa_0.7.17

```
Bootstrap: docker
From: ubuntu:18.10

%help

    Container for BWA 0.7.17
    http://bio-bwa.sourceforge.net/

%labels

    VERSION "BWA 0.7.17"


%post

    # install dependencies via apt
    apt update
    apt install -y \
        bwa

%runscript

    exec /usr/bin/bwa "$@"


```

Description

None

Build Metrics

Size (MB):101

Tag: bwa_0.7.17

Commit: 3c62cd682cafcac61867e6d8361bb413d67e6f37

Branch: master

Version (file hash) :eb49586488ed8fa5ba71da34f2fabc1a

Build Time: 40 seconds

Version 2.5.0-feature-squashbuild-secbuild-2.5.0.gddf62fb

Version 2.5

singularity-hub.org

Some software can’t go in a container

- Licensing issues *e.g.*
    - http://www.cbs.dtu.dk/cgi-bin/sw_request?rnammer
    - Can’t distribute the RepeatMasker DB

Workflow managers support containers and clusters

```
rule 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}’

rule 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}’
```

Cluster execution, *e.g.*:

```
snakemake --drmaa ” -q username” -j 32
```

Reproducible analysis stack

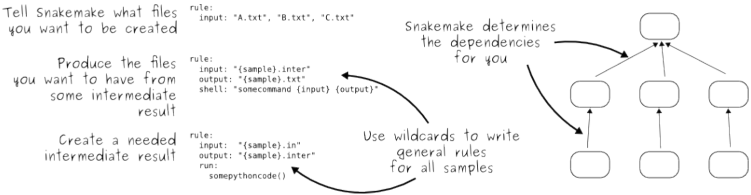
For reproducible bioinformatics:

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

`chmod 444 raw_reads.fastq` ?

Workflow manager (`snakemake`, `nextflow`)
+ VCS (`git`)

Software containers (`Singularity`)



Tell Snakemake what files you want to be created

```
rule:
  input: "A.txt", "B.txt", "C.txt"
```

Produce the files you want to have from some intermediate result

```
rule:
  input: "{sample}.inter"
  output: "{sample}.txt"
  shell: "somecommand {input} {output}"
```

Create a needed intermediate result

```
rule:
  input:  "{sample}.in"
  output: "{sample}.inter"
  run:
    somepythoncode()
```

Use wildcards to write general rules for all samples

Snakemake determines the dependencies for you

snakemake.readthedocs.io

Pain points of reproducible genomics

- Slow initially
- Convince the sysadmins to install Singularity
- Getting software in containers
- Duplication of effort

Who cares / why

- most of the time you are the only one who reproduces your results
- bonus to containers is easy installation / portability

Getting started

Reproducibility for bioinformatics:

- online lectures e.g. Adam Labadorf of Boston Uni

Workflow managers:

- Snakemake Tutorial
- Nextflow: Get started

Software containers:

- Singularity Quick Start

Version control:

- memorise a handful of `git` commands

These slides: https://github.com/TomHarrop/eresearch2020