

# Reproducible Computational Research

Mark Stenglein, MIP 280A4

# Components of reproducible computational research

- Data sharing
- Method documentation and transparency
- Code sharing and documentation
- Code portability and reproducibility
- Code version control
- Software dependency management

# Some good resources

Pat Schloss



*[https://riffomonas.org/reproducible\\_research/](https://riffomonas.org/reproducible_research/)*

OPEN  ACCESS Freely available online

 PLOS COMPUTATIONAL  
BIOLOGY

Editorial

## Ten Simple Rules for Reproducible Computational Research

Geir Kjetil Sandve<sup>1,2\*</sup>, Anton Nekrutenko<sup>3</sup>, James Taylor<sup>4</sup>, Eivind Hovig<sup>1,5,6</sup>

## Ten Simple Rules for Reproducible Computational Research

Geir Kjetil Sandve<sup>1,2\*</sup>, Anton Nekrutenko<sup>3</sup>, James Taylor<sup>4</sup>, Eivind Hovig<sup>1,5,6</sup>

Rule 1: For Every Result, Keep Track of How It Was Produced

Rule 2: Avoid Manual Data Manipulation Steps

Rule 3: Archive the Exact Versions of All External Programs Used

Rule 4: Version Control All Custom Scripts

Rule 5: Record All Intermediate Results, When Possible in Standardized Formats

Rule 6: For Analyses That Include Randomness, Note Underlying Random Seeds

Rule 7: Always Store Raw Data behind Plots

Rule 8: Generate Hierarchical Analysis Output, Allowing Layers of Increasing Detail to Be Inspected

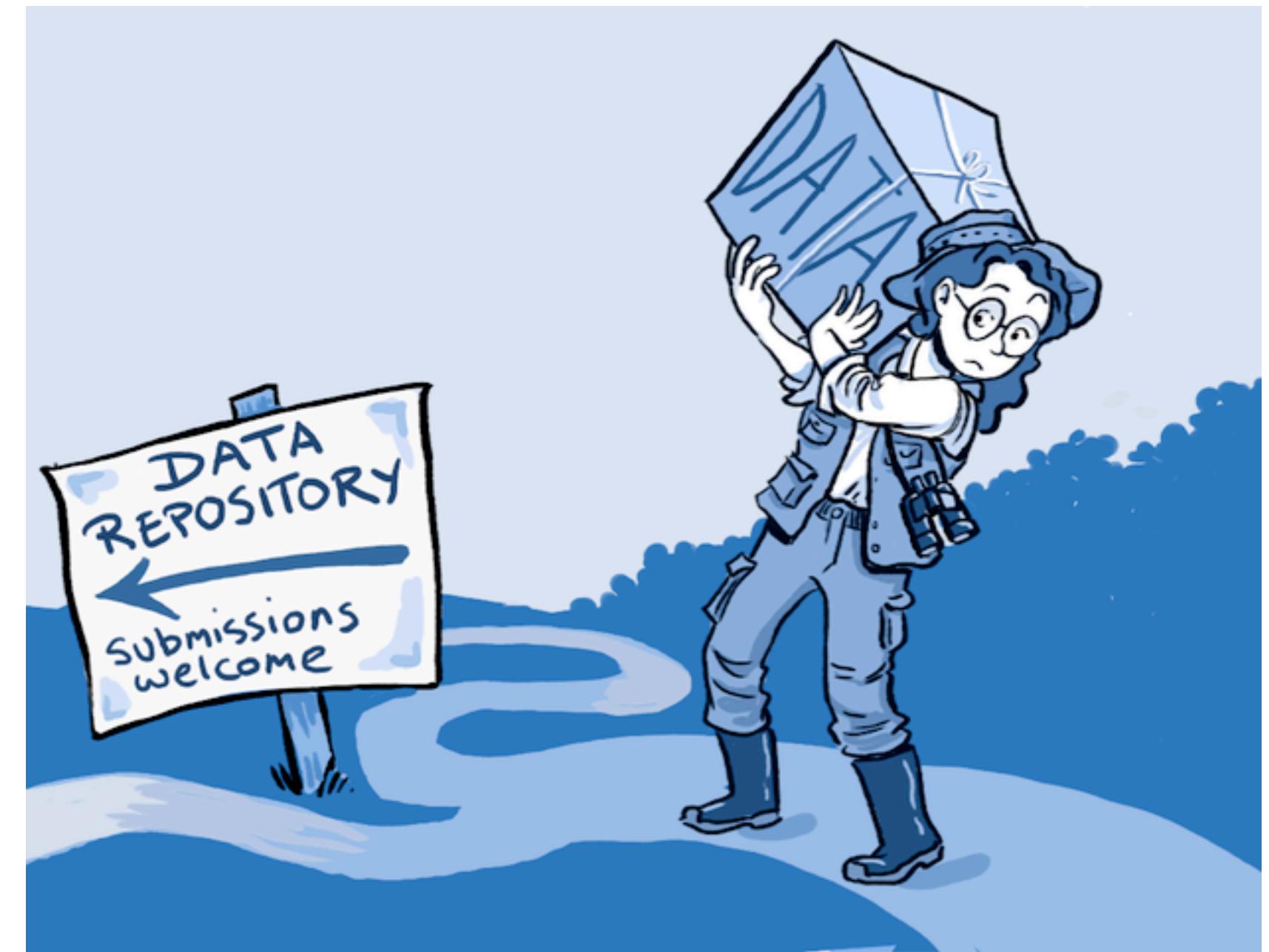
Rule 9: Connect Textual Statements to Underlying Results

Rule 10: Provide Public Access to Scripts, Runs, and Results

# Making your data available to others is critical

Best practices for data sharing:

- Deposit NGS data in the SRA database
- Deposit assembled sequences in Genbank or Assembly databases
- Put other important data (multiple alignments, tree files, etc.) in supplemental material and/or a Github repository



*Roche et al (2014) PLoS Biology*

Like in typical lab research, it is critical that you document what you are doing when performing computational analyses

In other words, keep a “computational lab notebook”



```
1 ---  
2 title: "R Notebook"  
3 output:  
4   html_document:  
5     df_print: paged  
6 editor_options:  
7   chunk_output_type: inline  
8 ---  
9  
10 This notebook is for the in vitro transcribed minigenome cocktail sequencing analysis.  
11  
12 ## Background/Purpose  
13 BSR T7/5 cells were transfected with Kairi L + N helper plasmids (T7 backbone) and 24 hours later transfected with IVT M segment at 24, 48, and 72 hours post-minigenome transfection. Harvested cells were split 80/20 for sequencing analysis and dual-luciferase sequencing analysis.  
14  
15 The IVT M segment minigenome cocktail contains:  
16 - ivt.mg. 1: Kairi M  
17 - ivt.mg.3: Kairi M negative  
18 - ivt.mg.4: Cache valley M  
19 - ivt.mg.6: Cache valley M negative  
20 - ivt.mg.11: Boarceia M  
21 - ivt.mg.13: Brazoran M  
22 - ivt.mg.14: Gamboa M  
23 - ivt.mg.15: LaCrosse M  
24 - ivt.mg.17: Oropouche M  
25  
26 ## Analysis  
27 ### Sequencing  
28 Pooled libraries were sequenced on the MiSeq with a v2 300 cycle nano kit with 2x151 reads and 2x8 bp indexes. The run directory /home/ngsdata/210901_M03520_0432_000000000-DCV9B  
29 ##### FASTQC  
30 Fastq files and analysis are on the cctsi-103 server here: /home/mllayton/analyses/2021_09_01_minigenome_ivt_cocktail  
31  
32 Reads were quality filtered and collapsed for PCR duplicate using the run_preprocessing_pipeline. Specifically, FASTQC and cutadapt  
33 ``{r}  
34 library(tidyverse)  
35 library(readxl)  
36 fastq_data <- read_excel("./fastqc_read_counts.xlsx")  
37 fasta_data  
38 fasta_data_tidy <- fasta_data %>%
```

Image: Katherine Stember, CC-BY-SA-4.0, [Link](#)

Computational lab notebooks can take many forms

- Handwritten, like a traditional lab notebook
- A word or google doc document
- A plain text file
- A “markdown” format file
- Shell scripts

It is more important that you keep track of what you did than how exactly you do so

The target audience of these documents are not necessarily other people.

You are often your own target audience. (How did I do that one thing 6 months ago?)

# Markdown format is a easy way to make nice-looking documents

## Markdown format is just a plain text format

```
## Variant calling exercise

MIP 280A4
---

### In this exercise, we will go through a variant calling pipeline.

The data we'll use was generated from a pool of Anopheles gambiae mosquitoes collected in Liberia. The dataset contains reads mapping to a virus (Anopheles flavivirus). We'll map the reads to the viral genome and use a variant caller to identify variants and determine their frequencies.

We will:

- download the files we need
- make a bowtie index from the viral genome (reference) sequence
- map reads to the reference sequence using bowtie2
- use lofreq to call variants
- visualize and inspect variants in vcf output file and in Geneious

### Download files needed for exercise from github

First, we need to download some files.

Open the terminal and change directory to a directory of your choosing (make a new directory if you'd like).

The files for this exercise are in the directory `/home/data_for_classes/2021_MIP_280A4/`. You need to copy this file into your current directory.

```
```

# Plain-text markdown can be rendered into a nice looking version

---

## Variant calling exercise

---

### MIP 280A4

---

**In this exercise, we will go through a variant calling pipeline.**

The data we'll use was generated from a pool of *Anopheles gambiae* mosquitoes collected in Liberia. The dataset contains reads mapping to a virus (*Anopheles flavivirus*). We'll map the reads to the viral genome and use a variant caller to identify variants and determine their frequencies.

We will:

- download the files we need
- make a bowtie index from the viral genome (reference) sequence
- map reads to the reference sequence using bowtie2
- use lofreq to call variants
- visualize and inspect variants in vcf output file and in Geneious

### Download files needed for exercise from github

First, we need to download some files.

# Markdown is simple plain text syntax that maps to nice-looking output formats

<https://www.markdownguide.org/basic-syntax/>

## Headings

To create a heading, add number signs (#) in front of a word or phrase. The number of number signs you use should correspond to the heading level. For example, to create a heading level three (<h3>), use three number signs (e.g., ### My Header).

| Markdown               | HTML                     | Rendered Output        |
|------------------------|--------------------------|------------------------|
| # Heading level 1      | <h1>Heading level 1</h1> | <b>Heading level 1</b> |
| ## Heading level 2     | <h2>Heading level 2</h2> | <b>Heading level 2</b> |
| ### Heading level 3    | <h3>Heading level 3</h3> | <b>Heading level 3</b> |
| #### Heading level 4   | <h4>Heading level 4</h4> | <b>Heading level 4</b> |
| ##### Heading level 5  | <h5>Heading level 5</h5> | <b>Heading level 5</b> |
| ###### Heading level 6 | <h6>Heading level 6</h6> | <b>Heading level 6</b> |

# Github is the de facto standard for sharing code (and data and lots of other things) these days

The screenshot shows the GitHub interface for the repository `stnglein-lab`. At the top, there's a navigation bar with links for Pull requests, Issues, Codespaces, Marketplace, and Explore. Below the navigation bar, the main content area features a large, circular sunburst chart on the left, which is a radial treemap visualization of the repository's structure. To the right of the chart, the repository name `stnglein-lab` is displayed. A descriptive text below the name states: "These are repositories for software and protocols created and used by the Stnglein Lab at Colorado State University". The central part of the page is titled "Popular repositories" and lists six public repositories:

- viral\_variant\_caller** (Public): A nextflow pipeline to call single nucleotide and structural variants in viral populations. Last updated 1 day ago. 7 stars.
- stnglein\_lab\_scripts** (Public): Stnglein Lab Scripts. Last updated 1 day ago. 5 stars.
- taxonomy\_pipeline** (Public): A set of scripts used to taxonomically assess the sequences in an NGS dataset. Last updated 1 day ago. 3 stars.
- 2017\_GDW** (Public): Documents and exercises used in the Genomics of Disease in Wildlife Workshop June 2017. Last updated 1 day ago. 3 stars.
- tick\_surveillance** (Public): Forked from laurenkleine/taxonomy-nf. Last updated 1 day ago.
- taxonomy-nf** (Public): Forked from laurenkleine/taxonomy-nf. Last updated 1 day ago.

At the bottom of the page, there are sections for "Customize your pins" and "Recent activity".

# This repository contains a pipeline to analyze NGS data

 [stnglein-lab/tick\\_surveillance](#) Public

[Pin](#) [Unwatch 1](#) [Fork 3](#) [Star 3](#)

[Code](#) [Issues 14](#) [Pull requests](#) [Discussions](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [...](#)

[main](#) [2 branches](#) [4 tags](#) [Go to file](#) [Add file](#) [Code](#)

 **stnglein-lab** Merge pull request #49 from stnglein... [...](#) ecbea6b 12 days ago [116 commits](#)

|                                                                                                            |                                                       |              |
|------------------------------------------------------------------------------------------------------------|-------------------------------------------------------|--------------|
|  conda                  | updated conda enviornment file, updated create fas... | last month   |
|  conf                   | test config                                           | 13 days ago  |
|  containers/singularity | fix comments re: venv creation                        | 20 days ago  |
|  input                  | Updated metadata template. Removed county filed ...   | 2 months ago |
|  lib/R                  | Removed use of custom singularity image (hosted o...  | 20 days ago  |
|  refseq                 | new surveillance column targets                       | 5 months ago |
| ...                                                                                                        |                                                       |              |

**About** 

Bioinformatics pipeline for analysis of amplicon sequencing of tick-associated microbes

[Readme](#) [MIT license](#) [3 stars](#) [1 watching](#) [3 forks](#)

**Releases** 4

# This repository (repo) contains code and data needed to make one figure

Screenshot of a GitHub repository page for "highlight\_residues\_on\_spike\_structure".

The repository was created by Mark Stenglein and has 4 commits. It contains 1 branch and 0 tags.

The repository files include:

- .gitignore
- 6VXX.afasta
- 6vxx.pdb
- [Figure\\_X\\_highlighted\\_re...](#)
- Figure\_X\_highlighted\_re...
- LICENSE
- README.md
- chimera\_commands.cmd
- generate\_chimera\_comm...
- residues\_to\_highlight.csv

The repository was last updated on Feb 28, 2021, at da2f119.

**About**

Highlight structures on SARS-CoV-2 spike structure using UCSF chimera

**Readme**

**MIT license**

**1 star**

**1 watching**

**0 forks**

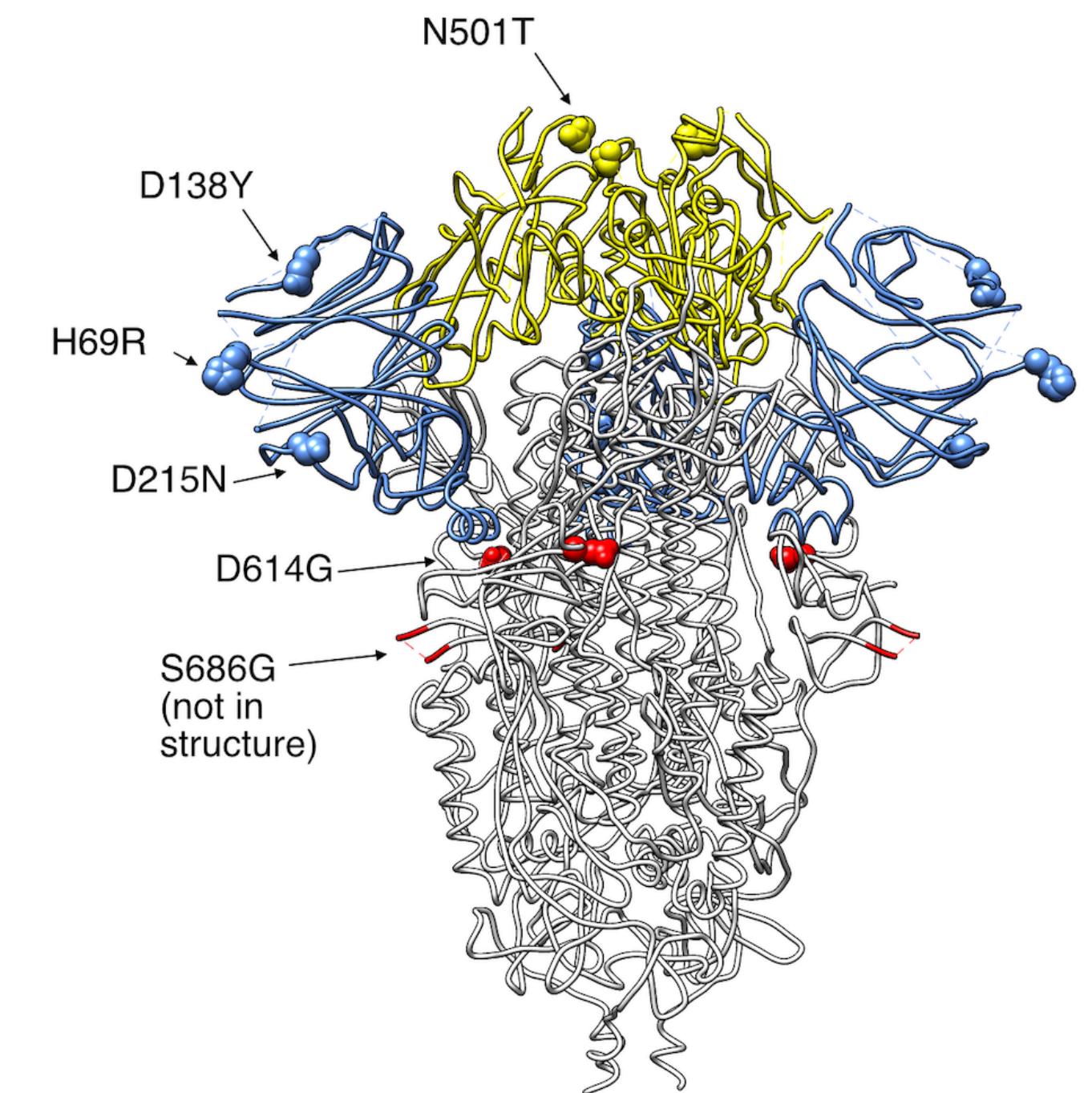
**Releases**

No releases published

[Create a new release](#)

**Packages**

No packages published



Bashor et al (2021)

# This repository contains a bunch of scripts

 [stnglein-lab / stnglein\\_lab\\_scripts](#) Public Pin Unwatch 1 Fork 1 Star 5

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

master 1 branch 1 tag Go to file Add file <> Code About ⚙

**Mark Stnglein update nr/nt db** 88e8ec6 on Feb 16 73 commits

|                                                                                                            |                                     |             |
|------------------------------------------------------------------------------------------------------------|-------------------------------------|-------------|
|  Histogram.pm           | added a bunch                       | 6 years ago |
|  LICENSE                | Initial commit                      | 7 years ago |
|  Levenshtein.pm         | added a bunch                       | 6 years ago |
|  README.md              | readme                              | 6 years ago |
|  acc_to_taxid.pm        | switch to using accessions from GIs | 5 years ago |
|  analyze.sam_error_mode | added a bunch                       | 6 years ago |

**Stnglein Lab Scripts**

-  [Readme](#)
-  [MIT license](#)
-  [5 stars](#)
-  [1 watching](#)
-  [1 fork](#)

---

**Releases** 1

 [First release of this repository](#) Latest

# This repository contains the exercises we've been doing in this course

The screenshot shows a GitHub repository page for 'stnglein-lab/MIP\_280A4\_Fall\_2022'. The repository is public, has 1 branch, and 0 tags. The 'Code' tab is selected. The repository contains files like 'conda\_environment', 'exercises', 'LICENSE', and 'README.md', all committed 2 days ago. The 'About' section describes it as course material for MIP\_280A4, Microbial Sequence Analysis, for Fall 2022, under a CC-BY-SA-4.0 license.

stnglein-lab / **MIP\_280A4\_Fall\_2022** Public

Pin Unwatch 1 Fork 0 Star 0

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 1 branch 0 tags Go to file Add file <> Code

**About**

Course Material for MIP\_280A4, Microbial Sequence Analysis, for Fall 2022

Readme CC-BY-SA-4.0 license

0 stars 1 watching 0 forks

| File              | Description       | Time        |
|-------------------|-------------------|-------------|
| conda_environment | download exercise | 27 days ago |
| exercises         | 2022 update       | 2 days ago  |
| LICENSE           | initial commit    | 27 days ago |
| README.md         | Initial commit    | 27 days ago |

A feature of git repositories is that they track changes to files (version control)

2022 update

[Browse files](#)

main

Mark Stenglein authored and Mark Stenglein committed 2 days ago 1 parent cb06113 commit a3c1427c84615ba41cb9a089efeae73979297639

Showing 1 changed file with 6 additions and 3 deletions.

Split Unified

exercises/variant\_exercise.md

@@ -28,7 +28,7 @@ Open the terminal and change directory to a directory of your choosing (make a n

28 The files for this exercise are in the directory `/home  
/data\_for\_classes/2021\_MIP\_280A4/`. You need to copy this  
file into your current directory.

29

30 `

31 - cp /home/data\_for\_classes/2021\_MIP\_280A4  
/variant\_exercise\_files.tar.gz .

28 The files for this exercise are in the directory `/home  
/data\_for\_classes/2021\_MIP\_280A4/`. You need to copy this  
file into your current directory.

29

30 `

31 + cp /home/data\_for\_classes/2022\_MIP\_280A4  
/variant\_exercise\_files.tar.gz .

32 `

33

A collection of changes in one or more files is called a “commit”

In this commit, I updated a URL from 2021 to 2022

Workflow managers allow you to make portable, scalable, reproducible pipelines

Portable: run on different types of computers (laptop, local server, cloud)

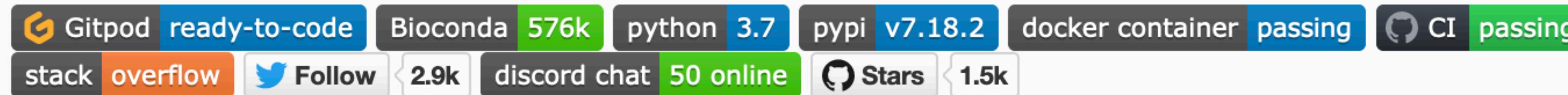
Scalable: run on one or many datasets or many many datasets

Flexible: change settings or variables (e.g. use a different reference sequence)

Reproducible: handle software dependencies smoothly

There are several popular workflow management systems

## Snakemake



The Snakemake workflow management system is a tool to create **reproducible** and **scalable** data analyses. Workflows are described via a human readable, Python based language. They can be seamlessly scaled to server, cluster, grid and cloud environments, without the need to modify the workflow definition. Finally, Snakemake workflows can entail a description of required software, which will be automatically deployed to any execution environment.

There are several popular workflow management systems

## Nextflow

### Data-driven computational pipelines

Nextflow enables scalable and reproducible scientific workflows using software containers. It allows the adaptation of pipelines written in the most common scripting languages.

Its fluent DSL simplifies the implementation and the deployment of complex parallel and reactive workflows on clouds and clusters.

# There are trade-offs to using workflows: additional work but big improvement in reproducibility

Part of a nextflow pipeline that runs fastqc on input fastq files

```
/*
Run fastqc on input fastq
*/
process initial_qc {
    label 'many_forks'
    label 'process_low'

    // singularity info for this process
    if (workflow.containerEngine == 'singularity' && !params.singularity_pull_docker_container) {
        container "https://depot.galaxyproject.org/singularity/fastqc:0.11.9--0"
    } else {
        container "quay.io/biocontainers/fastqc:0.11.9--0"
    }

    input:
    tuple val(sample_id), path(initial_fastq) from samples_ch_qc

    output:
    path("*.zip") into post_initial_qc_ch

    script:
    """
    fastqc $initial_fastq
    """
}
```

My advice:

Learn the basics of the tools and the command-line environment first

Then maybe start working with workflows for a pipeline you expect to use repeatedly

The actual fastqc command

Package managers (Conda) or Container systems (Docker, Singularity)

Deal with software dependencies and improve portability

**Conda**: create environments with one or more pieces of software and all of their dependencies. What we've been using in class (bio\_tools). Good reproducibility but there can be some issues (conflicts with already-installed R packages, for instance)

**Docker/Singularity**: Totally self-contained “container images” that have one or more software tools and their dependencies and an entire operating system. These are “virtual computers”. Very portable/reproducible: even better than conda.

There is good integration between workflow managers like nextflow and tools like conda or singularity to manage software dependencies

```
process MAKE_SEGMENT_TABLE {  
    tag "$tsv"  
    label 'process_low'  
  
    conda (params.enable_conda ? 'conda-forge::r-tidyverse=1.3.1' : null)  
    // why aren't singularity biocontainers updated to a newer tidyverse version?  
    container "${ workflow.containerEngine == 'singularity' && !task.ext.singularity_pull_docker_container ?  
        'https://depot.galaxyproject.org/singularity/r-tidyverse:1.2.1' :  
        'quay.io/biocontainers/r-tidyverse:1.2.1' }"
```

Some example nextflow code that includes information about docker packages or singularity containers that will be used.

# Let's make a repository!

## Outline

- Make GitHub logins
- Make a repository
- Clone it to Thoth
- Update the README file
- Commit the changes.

You will document your final project for this class in this repository! And it will be the basis for my grading.