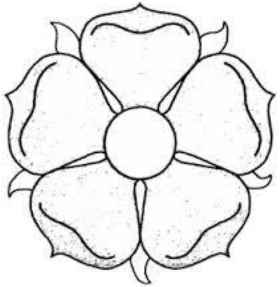
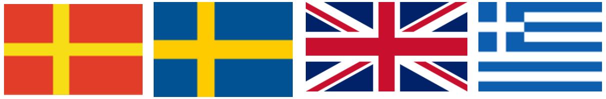
Best Practices in Handling Genomic Data



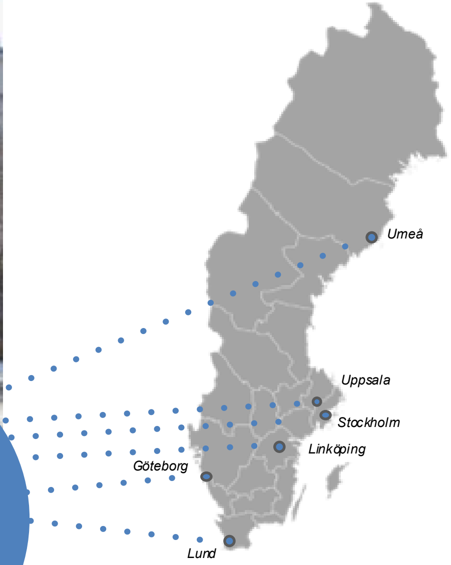
Dag Ahrén
Lund University Sweden



Cooking

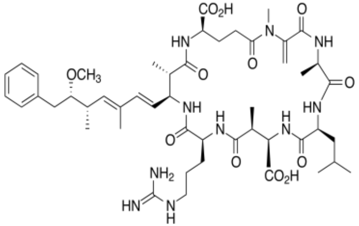
National Bioinformatics Infrastructure Sweden

~120 staff at six different sites across Sweden with expertise in many different omics-related areas

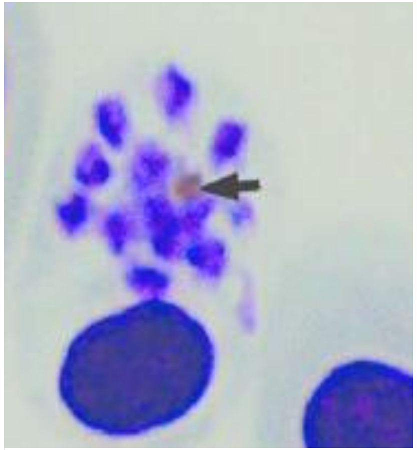


Genomic Ecology

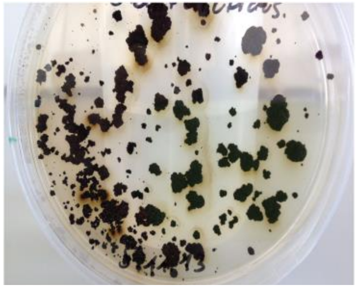
Single cell population genomics
Gonyostomum semen



Toxin gene clusters in *Microcystis*



Avian malaria host parasite
interactions



Adaptation to radiation in black
yeasts

Methanogens and methanotrophs in
SubArctic and Arctic Ecosystems

Genome project of the dung beetle
Kheper lamarcki

Tomas Larsson
Auguste de Pennart
Claudia Tocco
Marie Dacke
Dag Ahrén

NGI
Olga Vinnere Petterson
Christian Tellgren-Roth

LUND UNIVERSITY

IMPROBABLE
RESEARCH

Research that makes people LAUGH...then THINK

IG Nobel Prize
Marie Dacke,
Eric Warrant
Emily Baird

Let’s start cooking!

Ingredients

- Reproducible research
- Tools for reproducibility
- Special requests?
- Lab

Reproducible research, FAIR and Data management

Data management plan

https://dmponline.dcc.ac.uk/

Why important?

To be able to rerun analyses

Assist when publishing

Increase the usability of the data and results

. . .

Your future self will thank you!!!

My thoughts…

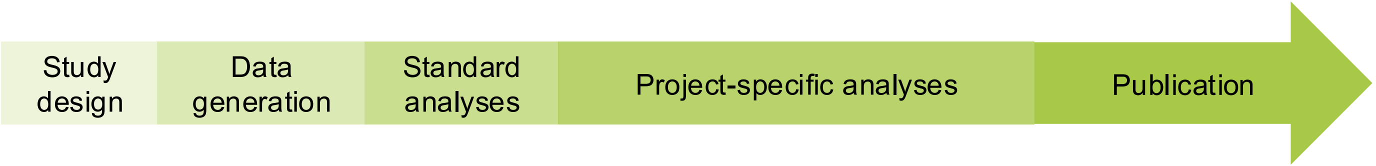
Set realistic goals

Share and help each other
& give positive feedback
(e.g. github repository)

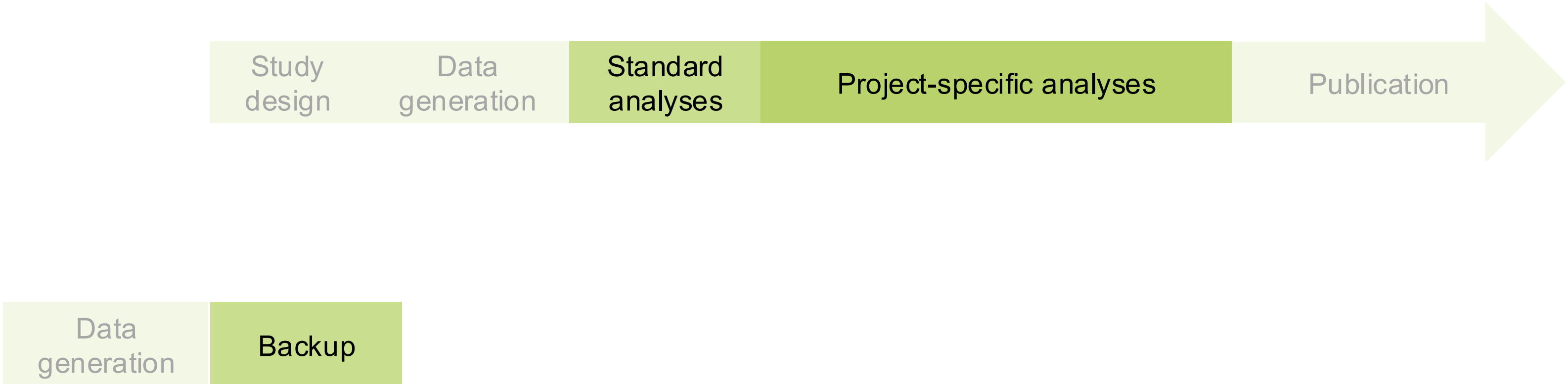
My goal today is to make
all of this a little bit easier!

Technical bits

Research Project Overview



Research Project Overview



Study design

Data generation

Standard analyses

Project-specific analyses

Publication

Data generation

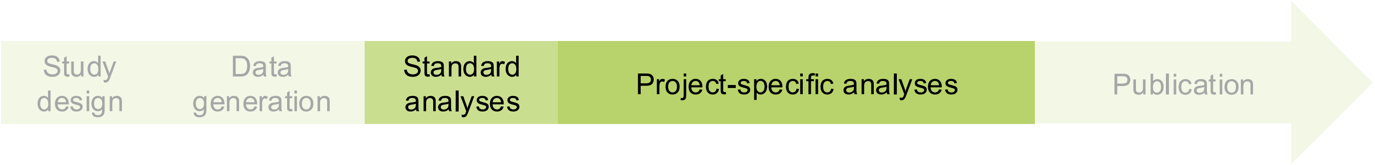
Backup

Backup

- Get an off-site backup for your raw data as soon as it arrives
- Make sure metadata is backed up with the raw data
- Once initial QC is complete, submit raw data to a data repository (with embargo)
- Get frequent backups of scripts
- Backup intermediate results

```
rsync -Pa
```

Research Project Overview



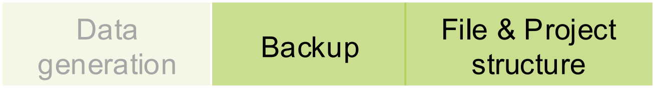
Study design

Data generation

Standard analyses

Project-specific analyses

Publication



Data generation

Backup

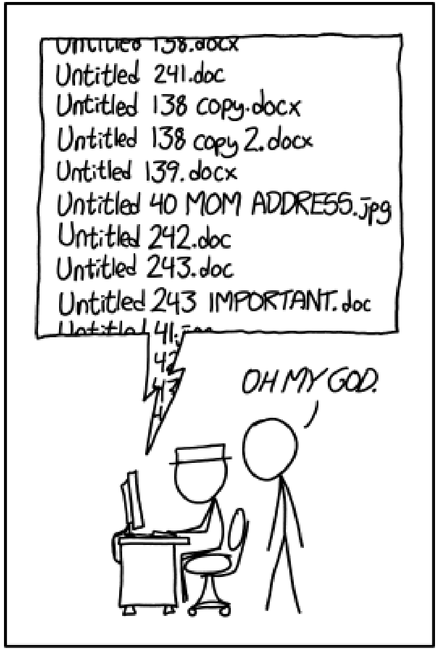
File & Project structure

Organise your project!

File names

- Use extentions to guide you (.txt .csv .fastq)
- Name files so that it is easy to understand and describe where it comes from (AT1_R1_trimmed.fq)
- Avoid any label that implies order relative to other files (Final1.txt UltraFinal.txt This_is_my_Final_Final_version2.txt)

File names



PROTIP: NEVER LOOK IN SOMEONE ELSE'S DOCUMENTS FOLDER.

My take on a strategy

(but with support from literature)

Totally fine if you have another strategy…

… but remember that chaos does not count as a strategy!!

Project

Good descriptive name of project, e.g ArcticMetagenome2025

- Include information about the goal and reasoning for the project *README*
- Data
- Analysis
- Docs
- Scripts
- Progs

Data

Read-only, raw data and meta data
> chmod -R 555 Data

This is an exact **COPY** of the data at the start of the project

Make a symbolic link to the raw data
Name the link something that is easy for you!

```
ln -s /data/runs/run42/SAMPLE_00123_L001_R1_001.fastq.gz \
     K_lamarcki_brain_sampleA_lane1_R1.fastq.gz
```

Note: Keep a backup at a separate location
Submit raw data to public repository early, with embargo

Docs

Put documentation (e.g R markdown, Quarto, Notes etc)

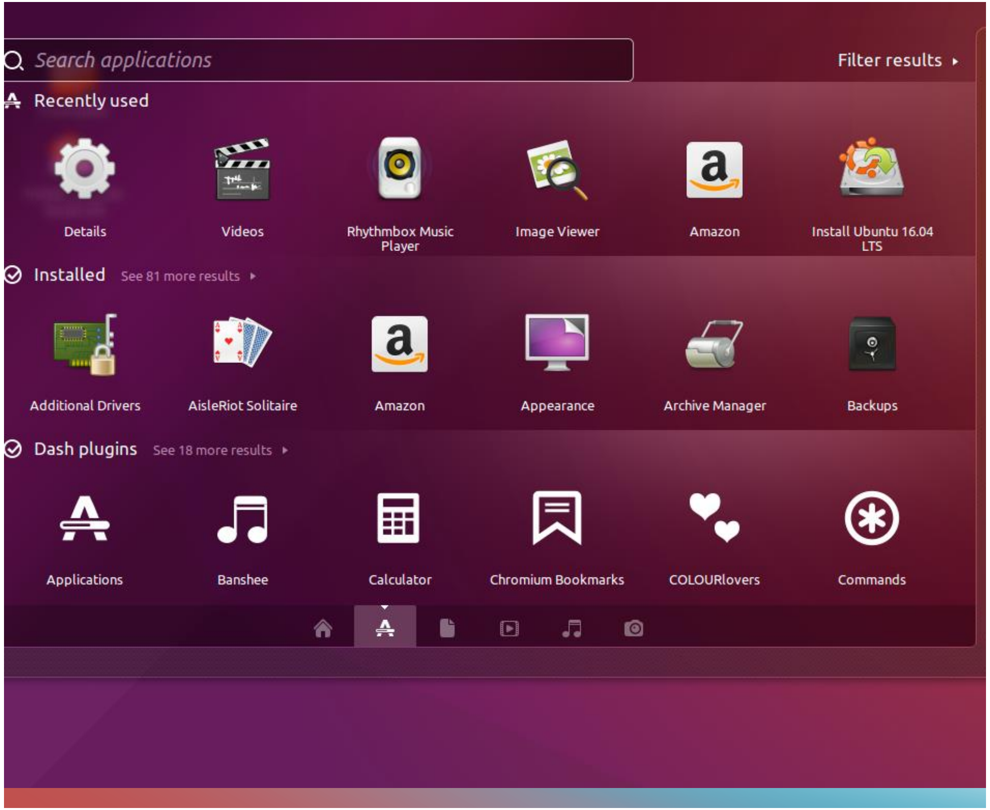
Scripts

Scripts, such as sbatch, bash, R scripts etc

Progs

Store software installed manually

Keep a record of software & versions

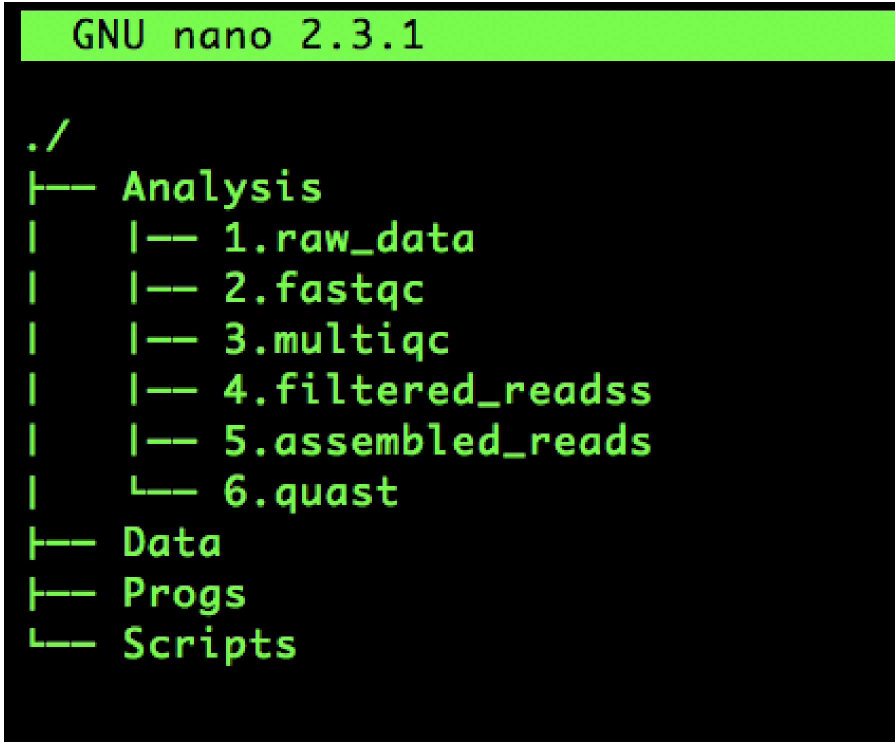


Analysis

Make a separate folder for each analysis.

1.raw_data is a symbolic link:

ln -s source destination



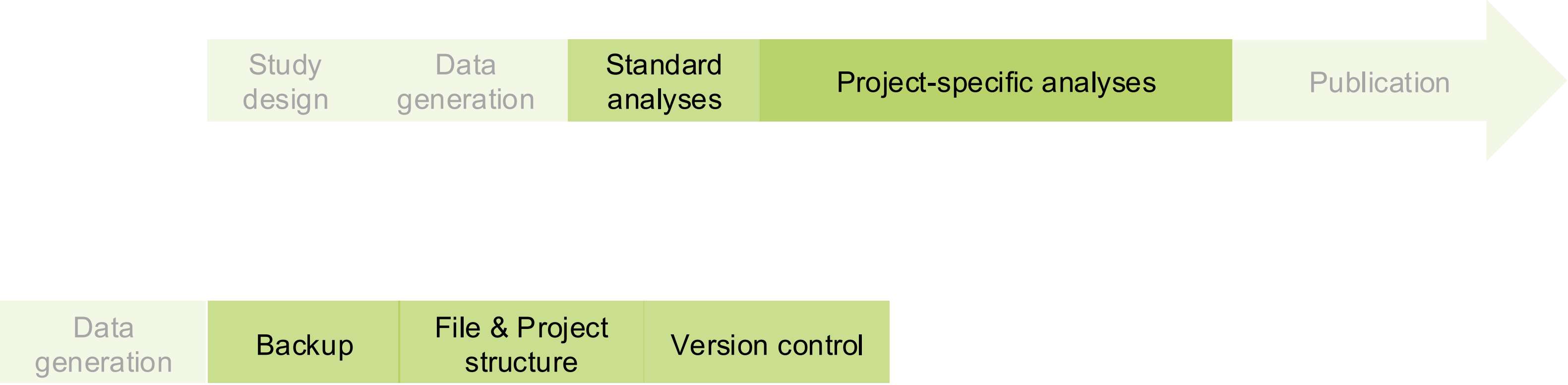
```
  GNU nano 2.3.1

./
├── Analysis
|    |── 1.raw_data
|    |── 2.fastqc
|    |── 3.multiqc
|    |── 4.filtered_readss
|    |── 5.assembled_reads
|    └── 6.quast
├── Data
├── Progs
└── Scripts
```

So you have a Project and File structure

Where do we go from here?

E362422 N6995173
Gamla Syle
16 Blåhammarens fjällstation via Enkälen
14 Storulvåns fjällstation via Spåime
Sylarnas fjällstation 2

Research Project Overview



Study design

Data generation

Standard analyses

Project-specific analyses

Publication

Data generation

Backup

File & Project structure

Version control

Work reproducibly

- Ten simple rules for Reproducible Computational Research (Sandve et al, 2013)

1. Track how results were produced (Quarto, Markdown, Juypiter notebook)
2. Avoid manual data manipulation
3. Archive/document all external software used. Versions!! (e.g. conda, R yml files)
4. Version control custom scripts (conda, markdown git/github)
5. Make it all available! (github)

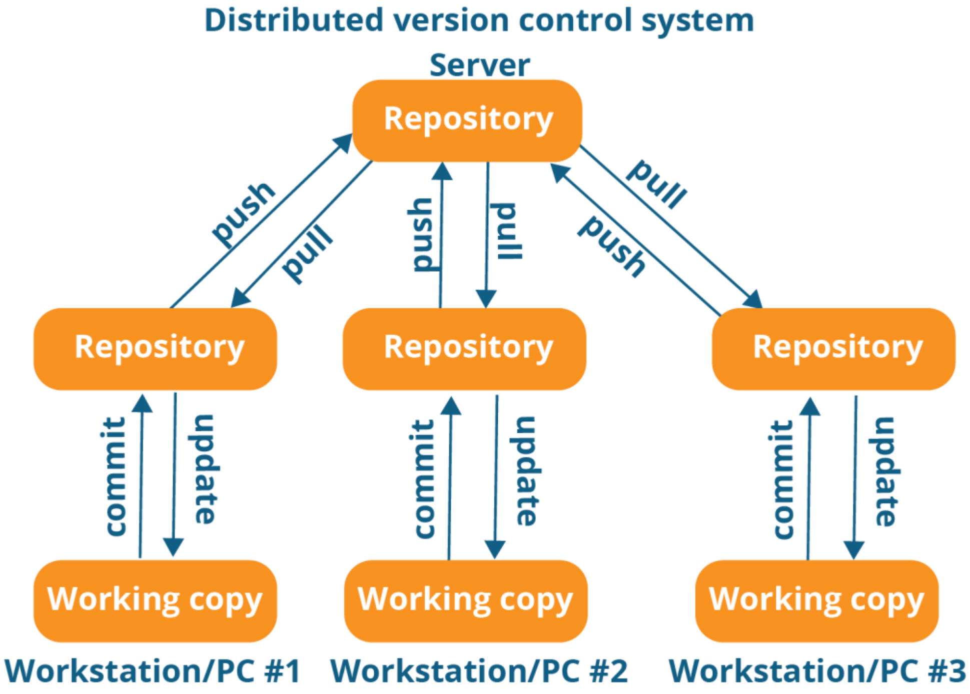
Version control

Git & Github

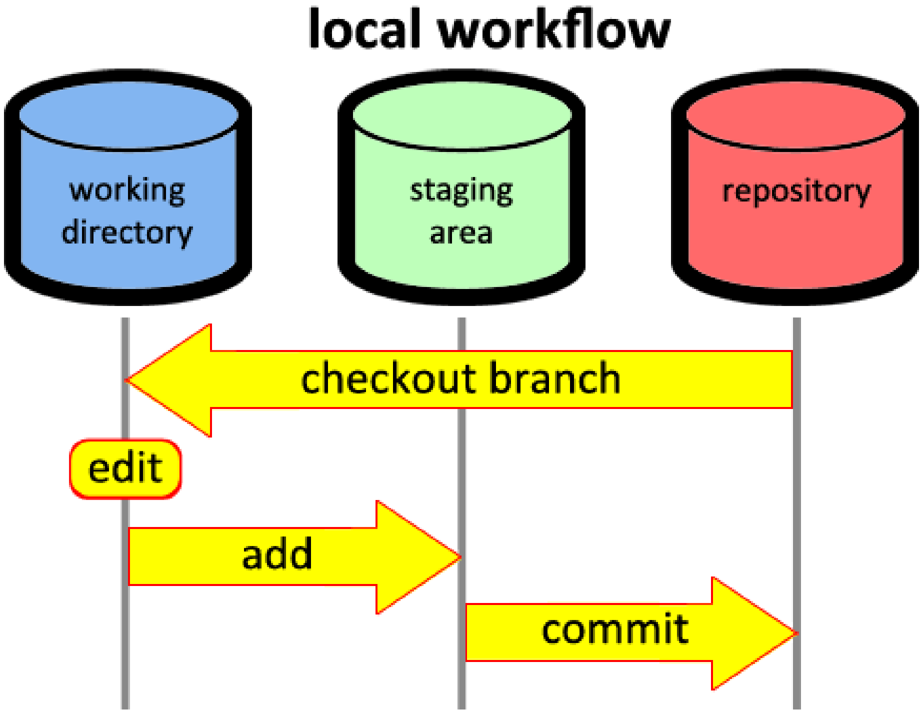
What is Git?
Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

What is GitHub?
GitHub is a web page were git repositories can be shared. It is a essentially social platform for code. Good for most things that fit with Git.

Git is distributed



Basic git workflow



Shortlist of the most useful terms in git

status

stage (add)

commit

push

pull

clone

branch

Recommendations when committing to the repository

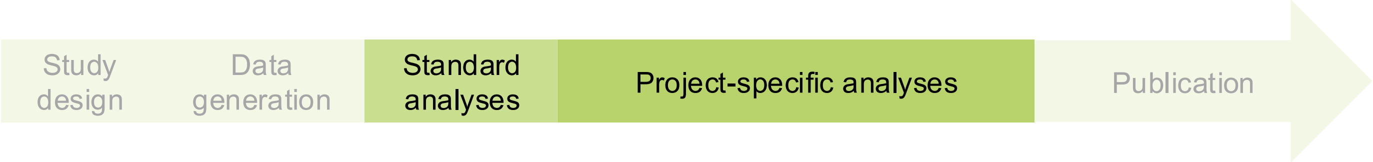
- Commit on a regular basis, ideally when one set of work has been performed and tested.
- Write short descriptive comments to each commit

Best practices when publishing

- Use git tag to tag a specific version that was submitted:

```
git tag ”submission1”
git switch -d submission1

git add config.yml
git commit -m "Increase number of reads”
git tag "revision-1"
```

Research Project Overview



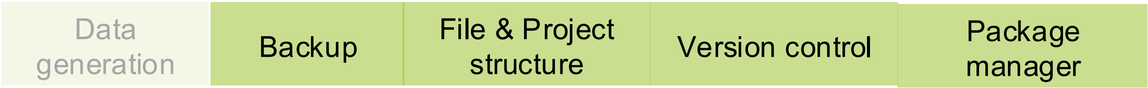
Study design

Data generation

Standard analyses

Project-specific analyses

Publication



Data generation

Backup

File & Project structure

Version control

Package manager

Conda

Package and environment manager

- Install software with dependencies
- Avoid dependency issues
- Save the software versions and dependencies in a file

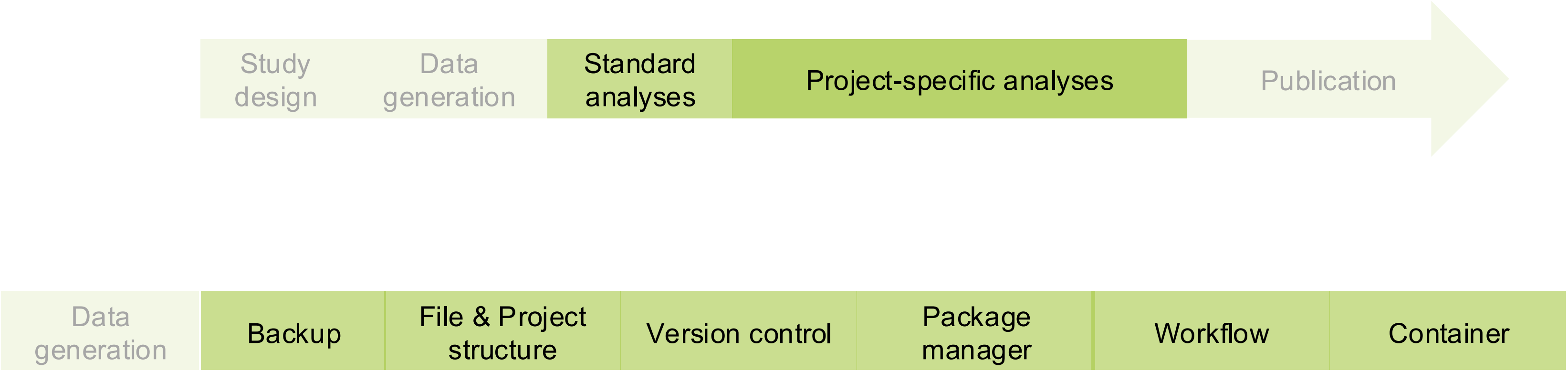
Conda commands

```
conda create -n project_A
conda env list
conda activate project_A
conda info –envs
conda install -c bioconda sra-tools
```

Save the environment software and dependencies to a file

```
conda env export > project_A_condaenv.yml
```

Research Project Overview



Study design

Data generation

Standard analyses

Project-specific analyses

Publication

Data generation

Backup

File & Project structure

Version control

Package manager

Workflow

Container

Other tools for reproducible science

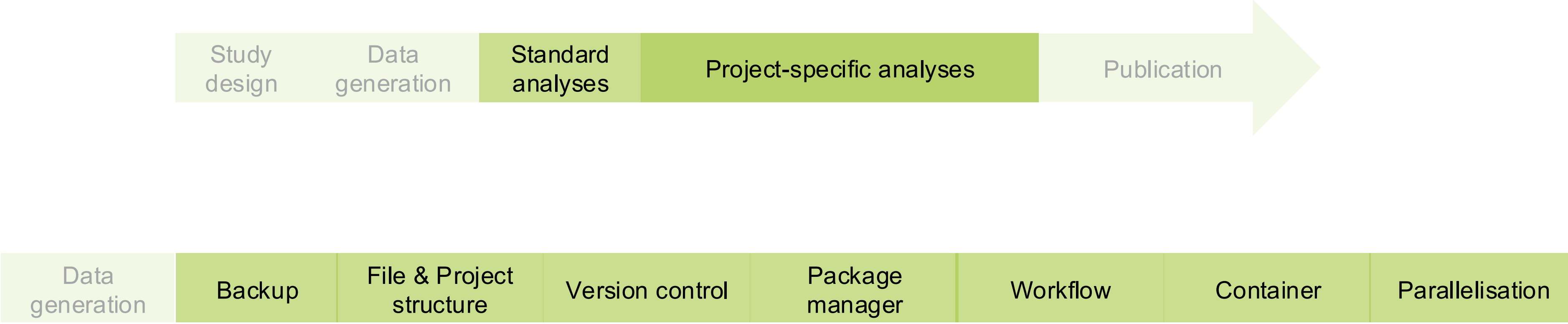
- Workflows such as Snakemake & Nextflow
- Containers Docker & Apptainer

Take home messages

Do not try to do all at once.

Start with file structure and backup.
then consider more advanced steps such at git and conda Set goals that are realistic

Research Project Overview



Study design

Data generation

Standard analyses

Project-specific analyses

Publication

Data generation

Backup

File & Project structure

Version control

Package manager

Workflow

Container

Parallelisation

Parallellization in genomics

Why is Parallelization Important?

- **Data Volume**: The sheer size of bioinformatics datasets, such as genomic sequences, requires robust computational approaches.
- **Complexity**: Many bioinformatics algorithms involve complex calculations that can benefit from parallel execution.
- **Time**: In time-sensitive research, reducing computational time can accelerate discovery and the application of findings.

Approaches to Parallelization

Multithreading: Utilizing multiple threads within a single processor to execute multiple tasks concurrently.

Distributed Computing: Spreading tasks across multiple compute nodes in a cluster or cloud environment.

GPU Acceleration: Using Graphics Processing Units (GPUs) for their parallel processing capabilities with large numbers of cores suited for certain types of calculations.

Not all software can be efficently parallelized

E.g Genome assembly Check if multithreading is an option

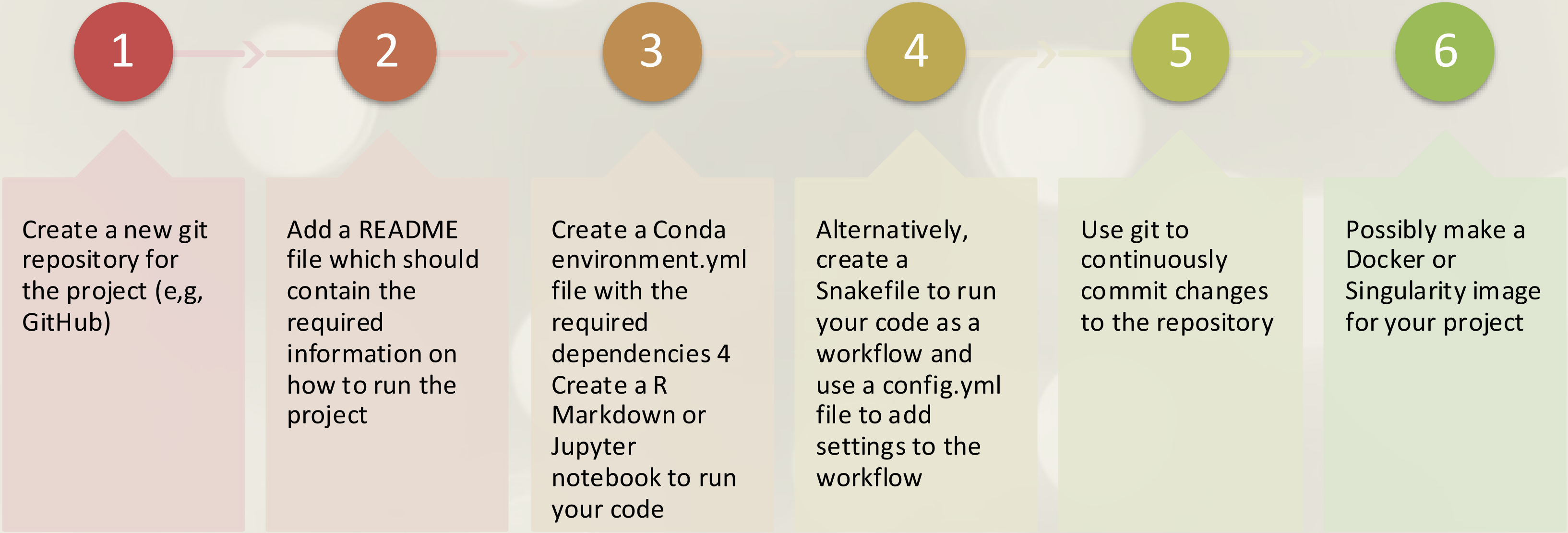
Eric slide on alignment

Tools & Libraries

- **GNU parallel**
- MPI (Message Passing Interface)
- OpenMP (Open Multi-Processing)
- Bioconductor packages (e.g., BiocParallel)

Pick your poison

Putting it all together



1. Create a new git repository for the project (e,g, GitHub)
2. Add a README file which should contain the required information on how to run the project
3. Create a Conda environment.yml file with the required dependencies 4 Create a R Markdown or Jupyter notebook to run your code
4. Alternatively, create a Snakefile to run your code as a workflow and use a config.yml file to add settings to the workflow
5. Use git to continuously commit changes to the repository
6. Possibly make a Docker or Singularity image for your project

Best Practices Lab

Lab on Git and Conda

NBIS Data management & Reproducibility courses

Setup on your instance

```
git clone https://github.com/NBISweden/workshop-reproducible-research.git
```

Avoid creating a repo inside another repo!

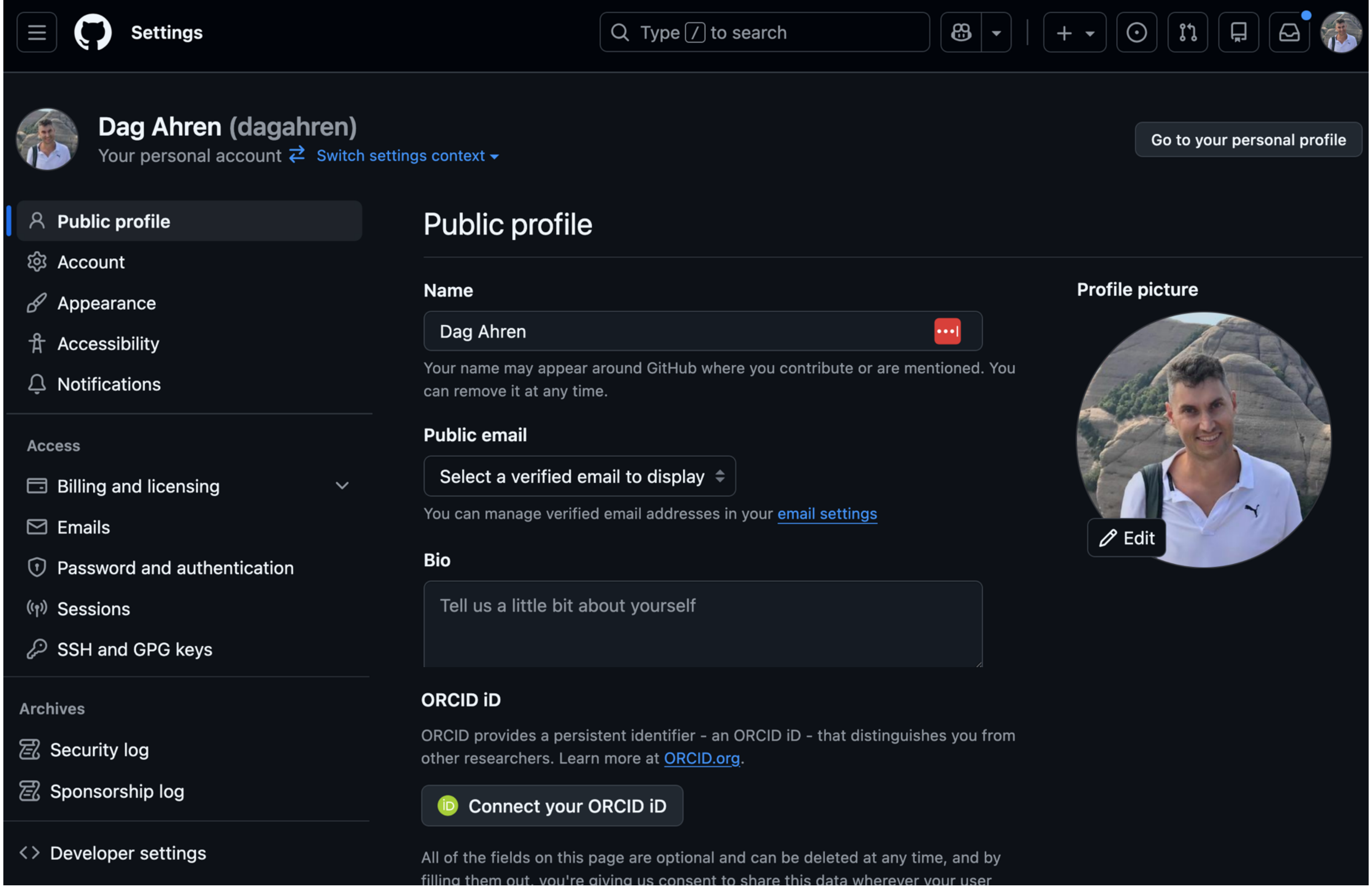
GitHub

From your GitHub account, go to Settings → Developer Settings → Personal Access Token → Tokens (classic) → Generate New Token (Give your password) → Fillup the form → click Generate token → Copy the generated

Token, it will be something like ghp_sFhFsSHhTzMDreGRLjmks4Tzuzgthdvfsrta

Add the copies token string and use as password

May need to do: git push -u origin main



Public profile

Name

Dag Ahren

Your name may appear around GitHub where you contribute or are mentioned. You can remove it at any time.

Public email

Select a verified email to display

You can manage verified email addresses in your email settings

Bio

Tell us a little bit about yourself

ORCID iD

ORCID provides a persistent identifier - an ORCID iD - that distinguishes you from other researchers. Learn more at ORCID.org.

All of the fields on this page are optional and can be deleted at any time, and by filling them out, you're giving us consent to share this data wherever your user

Thanks

I look forward to talk to you about:

- Reproducible research
- Different career paths
- Work-Life balance
- Life in Sweden/UK/Greece

… and Food!