EPS Get together 2019

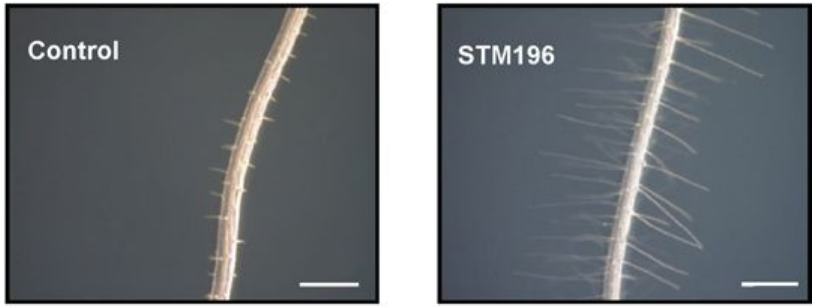
"*Homo biologicus informaticus*":
must know & good practices for the future Life Scientist

Marc Galland - Data Scientist/Manager (University of Amsterdam) & Founder BioData Services
www.mgalland.info
www.biodataservices.eu

Who am I - PhD

Montpellier (France)

Arabidopsis thaliana



Beneficial bacteria (PGPR) interaction with the ethylene hormonal pathway

Who am I - Postdoc (1)

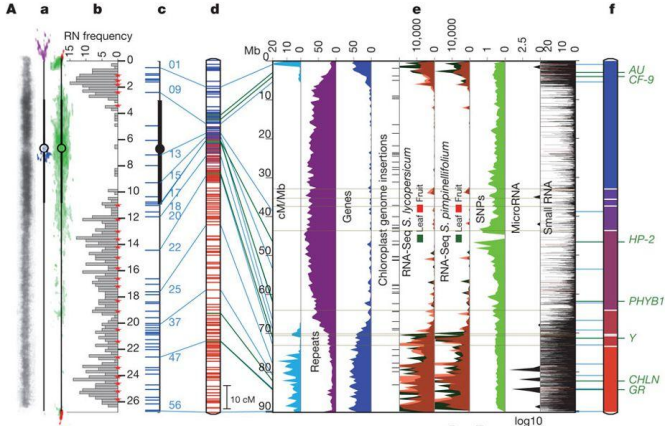
Versailles (France)

Post-transcriptional regulation of Arabidopsis & rice seed germination

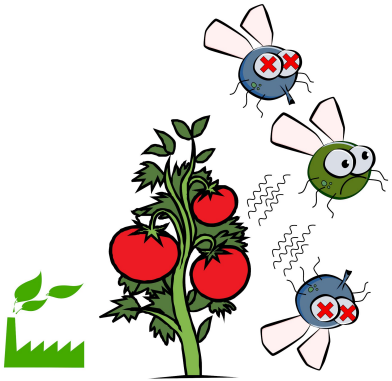
Partnership with industry

Who am I - Postdoc (2)

Amsterdam

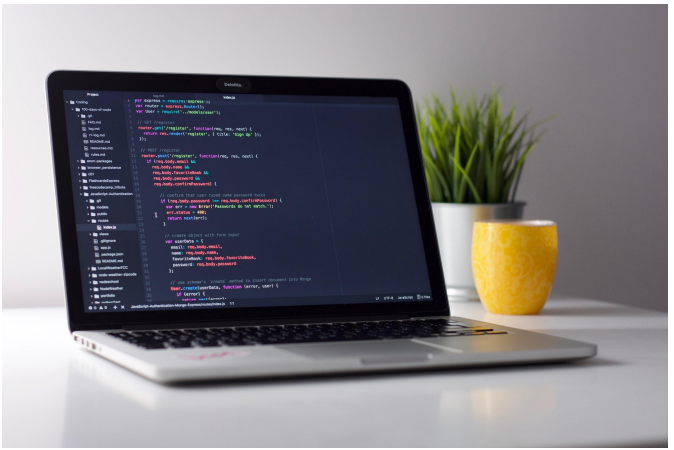


Tomato genomics



Finding genetic & metabolic determinants to insect pests

From *in vitro* to *in silico*



From the pipette to the computer

My take home messages

The modern biologist is more and more a wet-lab / dry-lab hybrid

- The “pipette biologist” versus “computer biologist” ? No…just life scientists!
- Every scientist is a data scientist: generates data, analyses them and make plots…

Good practices in programming are like good practices in the lab

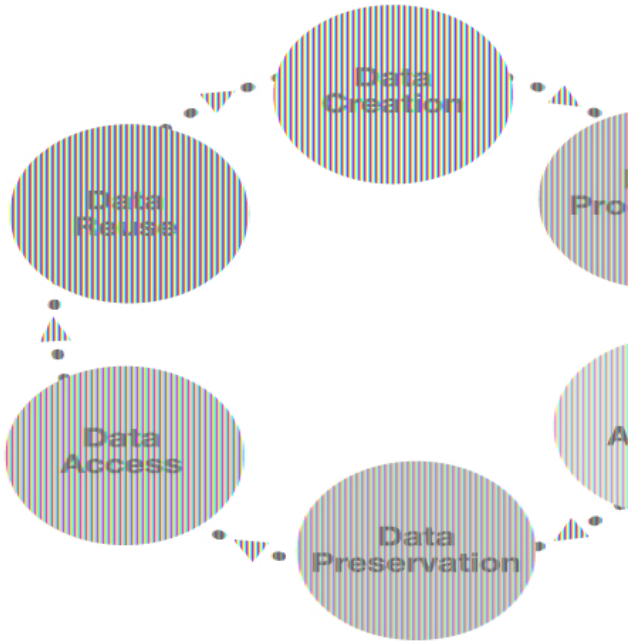
- Western Blot without gloves? At least you get proteins on your membrane…
- Nothing written in the laboratory notebook….well…

Outline

Photo by Pablo Merchán Montes on Unsplash

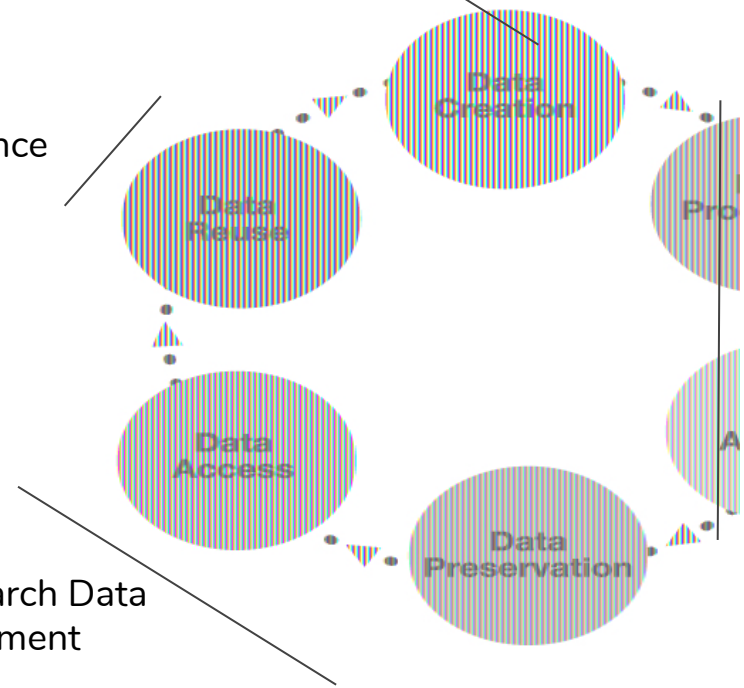
1. The data avalanche
2. Main programming tools useful in computational biology
3. Project management and organisation
4. Good practices in scientific programming
5. Research Data Management
6. Reproducible Research & Open Science
7. Building a local community of practice in programming & data science

Outline: the research data life cycle



Outline: the research data life cycle

1. Data avalanche



2. Tools (R,Python,Shell)

3. Project management

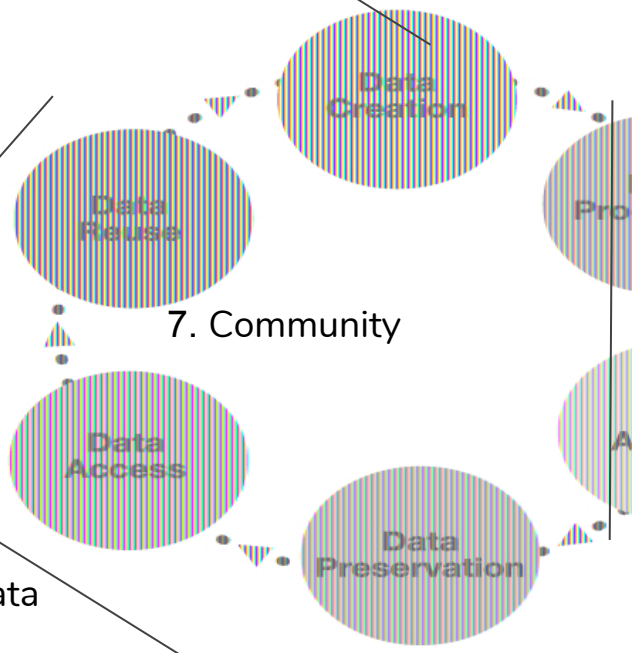
4. Good practices

5. Research Data Management

6. Open Science

Outline: the research data life cycle

1. Data avalanche



6. Open Science

2. Tools (R,Python,Shell)

3. Project management

4. Good practices

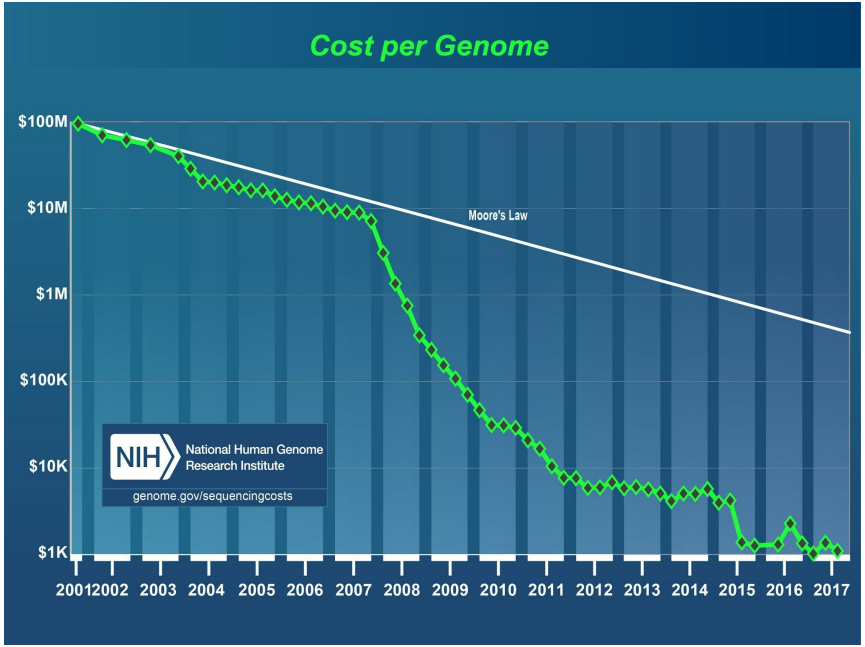
7. Community

5. Research Data Management

1. The data avalanche

Photo by Nicolas Cool on Unsplash

Genomic costs are plunging



https://www.genome.gov/27565109/the-cost-of-sequencing-a-human-genome/

Genomics vs other Big Data

Genomics
(various)

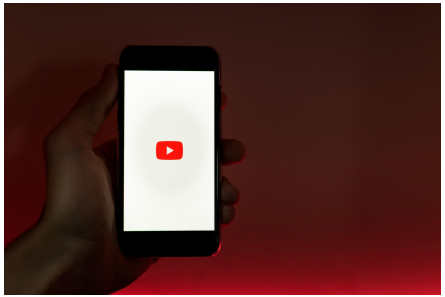


Photo by Szabo Viktor on Unsplash

YouTube
(video)

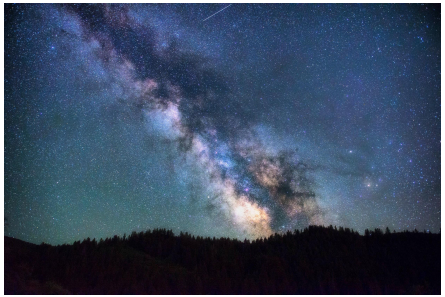


Photo by Christian Nielsen on Unsplash

Astronomy
(images)

Twitter
(text)

Quiz

Go to www.menti.com

Enter the code: 84 61 20

Quiz - answers (2015 estimates)

Genomics (NCBI SRA)

35 petabytes (PB) **per year**

1 PB = 10^{15} bytes

YouTube

240 petabytes (PB) **per day**.

1 PB = 10^{15} bytes

Twitter

547 terabytes (TB) **per day**

1 TB = 10^{12} bytes

Quiz - answers (2025 estimates)

Genomics (NCBI SRA)

1 zetabytes (ZB) per year

1 ZB = 10^{21} bytes

YouTube

~ 1340 - 2070 PB per year

1 PB = 10^{15} bytes

Twitter

1.4 PB per year

1 PB = 10^{15} bytes

But wait....we all generate data... every scientist is a data scientist :-)

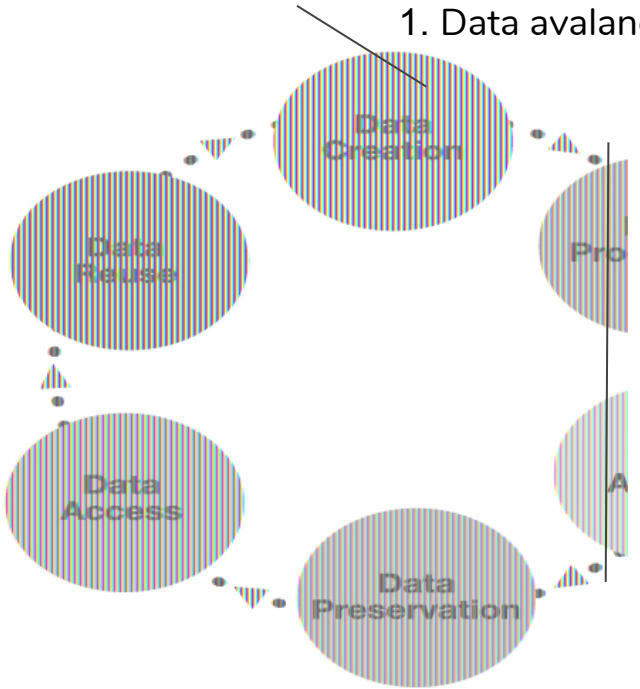
Phenotypes

Photo by Erwan Hesry on Unsplash

Lab experiments (e.g. qPCR)

Photo by Louis Reed on Unsplash

Outline: the research data life cycle



2. Tools (R,Python,Shell)

3. Project management

4. Good practices

2. Programming tools useful in computational biology

Photo by Hunter Haley on Unsplash

What is (scientific) programming?

A problem to solve

Every program begins with a problem you want to solve.

A solution to the problem

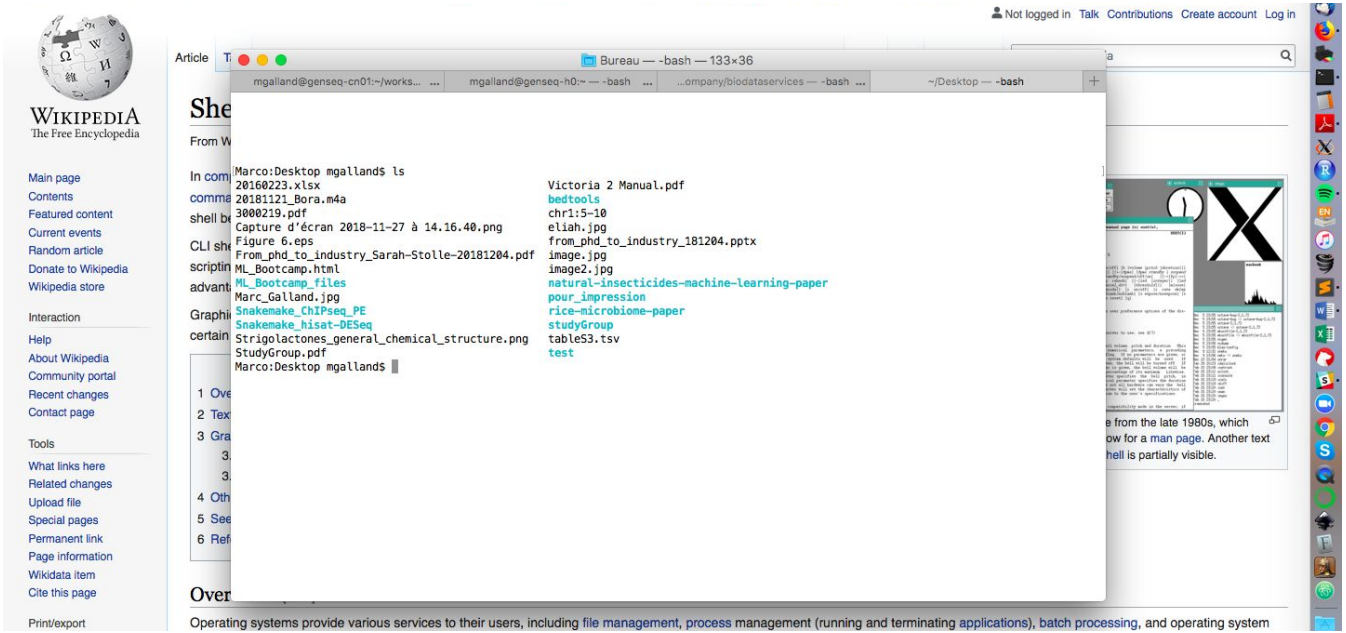
The solution to that problem is referred to as an algorithm.

The solution translated into a computer language

And finally, that solution is translated into a programming language, like Python, that the computer can understand.
That package of code that's run on a computer is called a **program**.

Photo by Paul Felberbauer on Unsplash

A Swiss Army Knife: the Shell



https://en.wikipedia.org/wiki/Shell_(computing)

Jack of all trades: the Shell

Demo time!

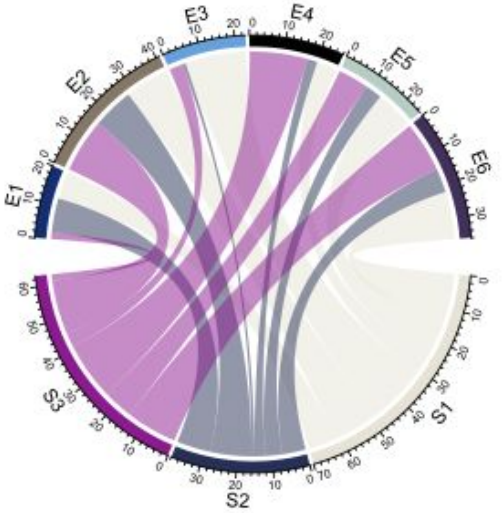
- List files in a directory on your computer
- Copy files
- Connect to the LISA cluster (SURFsara)

R and RStudio

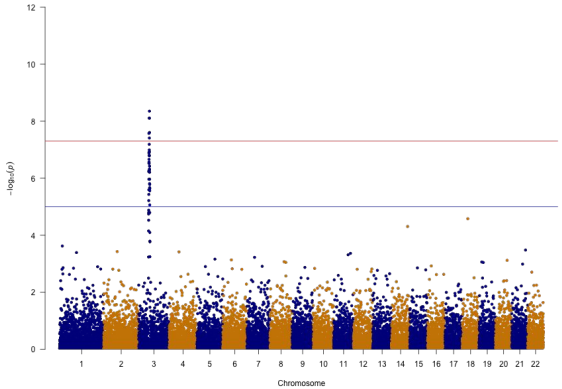
- R is a free software environment for statistical computing and graphics.
- RStudio is an Integrated Development Environment (IDE) that helps to develop R programs
- Helps with:
  - **Data transformation:** transposing a table, filtering row values based on a criteria. -- tidyverse
  - **Set operations:** A union B, A intersection B, join operations. -- UpSetR
  - **Statistics:** hypothesis testing (t-test, ANOVA), model fitting (regressions), ... -- base
  - **Exploratory Data Analysis:** Principal Component Analysis, heatmaps, clustering -- ggbiplot
  - **Plotting:** publication-grade plots with ggplot2 -- ggplot2
  - **Genomics:** Genomic data analysis with packages from the Bioconductor channel

(Bioconductor) Huber et al. (2015) *Nature Methods* **12:**115–121

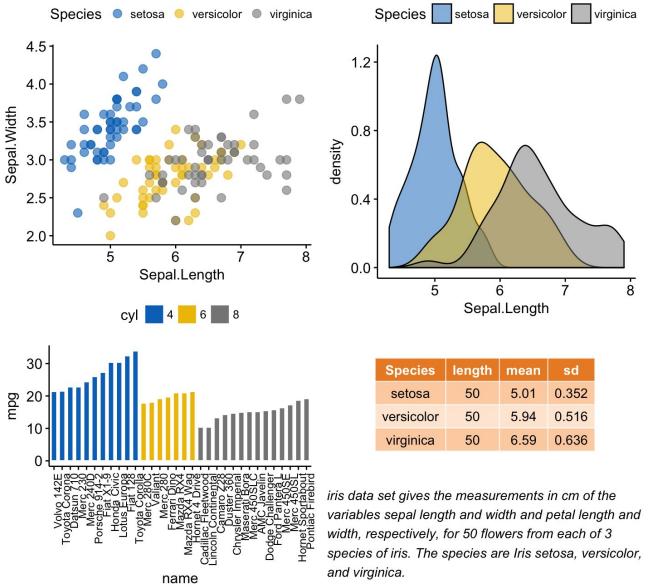
R and RStudio gallery



Circular representation of relationships



GWAS (Manhattan plot)



Species	length	mean	sd
setosa	50	5.01	0.352
versicolor	50	5.94	0.516
virginica	50	6.59	0.636

iris data set gives the measurements in cm of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris. The species are Iris setosa, versicolor, and virginica.

Scatter plots, density plots, etc.

Gu, Z. (2014) circlize implements and enhances circular visualization in R. Bioinformatics. DOI: 10.1093/bioinformatics/btu393
Turner, S.D. (2014) qqman: an R package for visualizing GWAS results using Q-Q and manhattan plots. *bioRxiv* DOI: 10.1101/005165

R and RStudio

Demo time

3 tasks

- **Transforme** a dataset into the tidy format -- dplyr package
- **Explore** the dataset using plots -- ggplot2 package
- **Filter** the dataset -- dplyr package

Python

A bit of Dutch prideness! Guido van Rossum invented Python in the early 90s' at the Centrum Wiskunde Informatica (Amsterdam).

Python 0.9.1 was released in... 1991 !

Many modules ("packages") built over Python. Well active community & development

A dedicated scientific Python distribution well adapted for scientists (i.e. Anaconda)

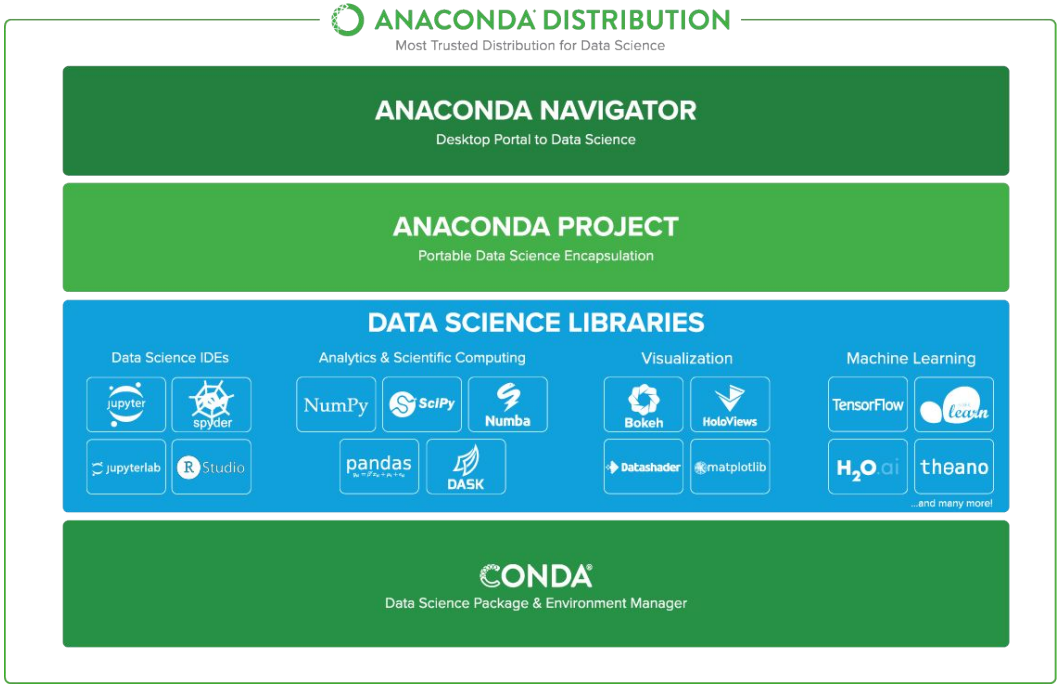
Anaconda distribution

Useful for scientific computing:
- data science
- machine learning applications
- large-scale data processing

Conda package management system.

Over 6 million users

1400 popular data-science packages suitable for Windows, Linux, and MacOS.

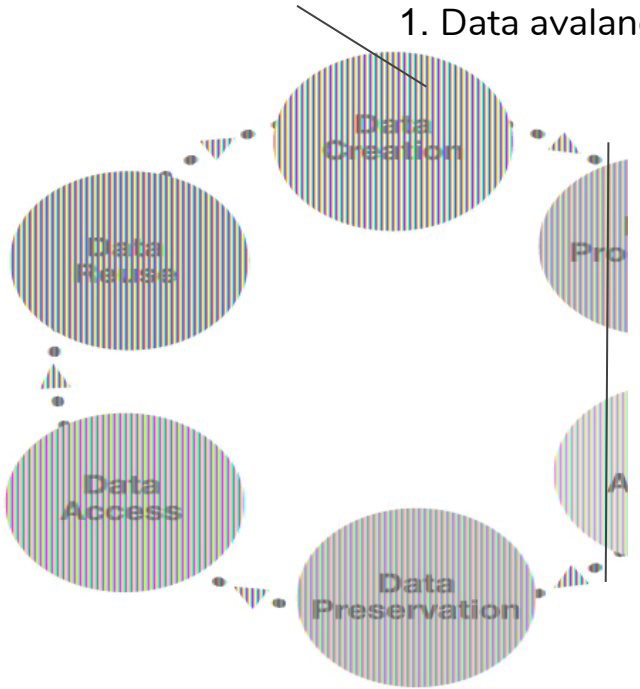


Using the Anaconda distribution

Demo time!

- **Start** the Anaconda navigator
- **Load** a Jupyter notebook
- **Make** a plot

Outline: the research data life cycle



2. Tools (R,Python,Shell)

3. Project management

4. Good practices

3. Project management

Photo by Daniele Riggi on Unsplash

Project management

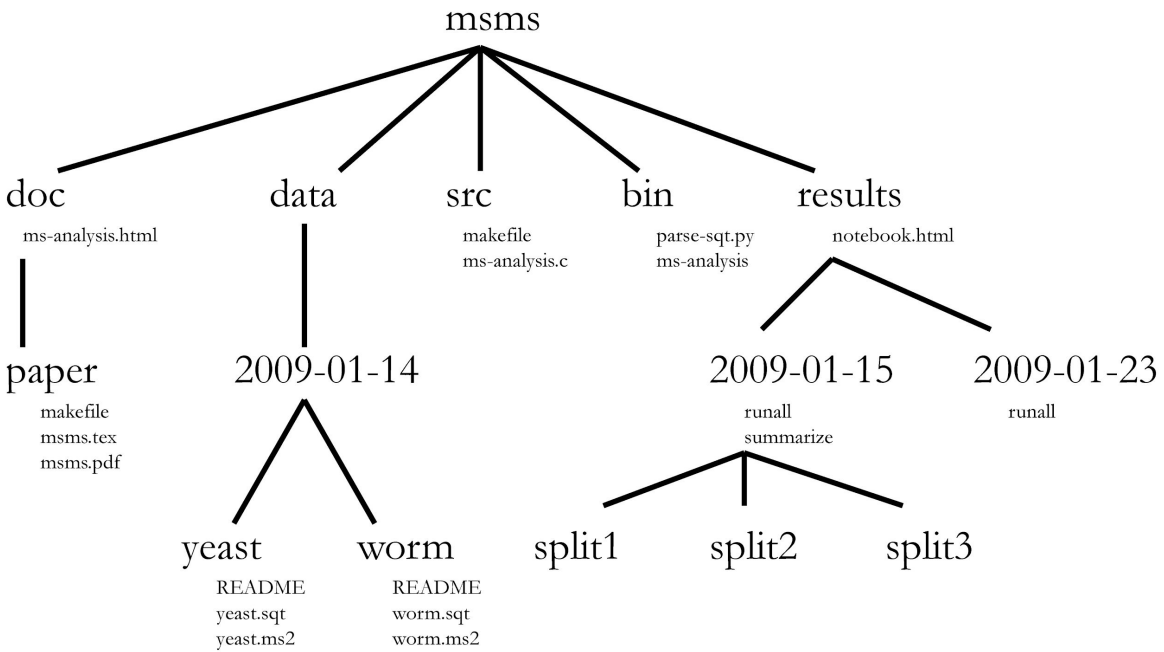
The core guiding principle is simple: **someone unfamiliar with your project should be able to look at your computer files and understand in detail what you did and why.**

This ‘‘someone’’ could be:

- someone who read your published article and wants to try to reproduce your work
- a colleague who wants to understand the details of your experiments
- a student working in your lab who wants to extend your work after you have moved on to a new job
- your research supervisor, who may be interested in understanding your work
- **Most commonly, however, that ‘‘someone’’ is you = your future self = you in 6 months**

A Quick Guide to Organizing Computational Biology Projects Noble WS (2009) A Quick Guide to Organizing Computational Biology Projects. PLOS Computational Biology 5(7): e1000424. https://doi.org/10.1371/journal.pcbi.1000424

Project management - inspiration



A Quick Guide to Organizing Computational Biology Projects Noble WS (2009) A Quick Guide to Organizing Computational Biology Projects. PLOS Computational Biology 5(7): e1000424. https://doi.org/10.1371/journal.pcbi.1000424

Project management - inspiration

Project root (name = grant number for instance or project name e.g. “NWO-12826”)

README.txt >>> File that gives some explanations about the project

- 01_Project info
  - 01. grants
  - 02. internship_proposals
  - 03. reports
  - ...
- 02_Experiments
  - 01.raw_data >>> only read permissions. Files should not be changed
  - 02.data_analysis >>> intermediate files
  - 03.protocols >>> for lab experiments
  - ...
- 03_Code >>> equivalent to protocols but for the dry lab
  - Github_repository_number_1 >>> your scripts maintained using a version control system
  - Github_repository_number_2 >>> another folder with other useful scripts
  - ...
- 04_Dissemination
  - 01.presentations
  - 02.publications
  - 03.patents
  - 04.outreach >>> a lecture in a primary school, a science fair...
  - ...

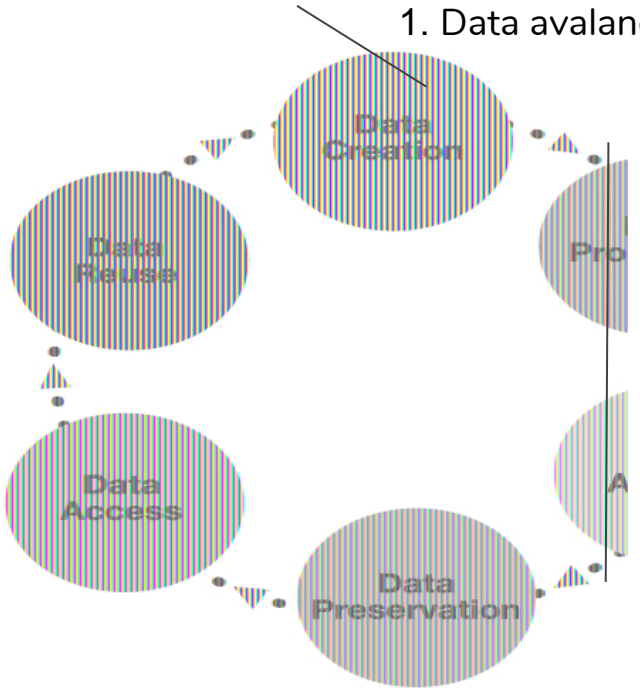
Project management - must do

- Write dates in YYYY-MM-DD = 2019-02-11 so that folders get ordered correctly
- Use two digits 01 02 03 and not 1 2 3 otherwise 12 gets before 2
- No spaces in file names or folders (because Shell does not like them)

My own way

Name	Date Modified	Date Created	Size	Tags	Date Last Opened	Date Added
00.manuscripts	08/11/2018, 12:02	11/08/2017, 10:34	12.36 GB	--	23/11/2018, 14:42	05/10/2017, 14:21
01.bibliography	20/11/2018, 13:04	19/09/2017, 14:44	412.4 MB	--	--	05/10/2017, 14:21
02.grants	18/06/2018, 11:00	19/09/2017, 14:44	79.4 MB	--	--	05/10/2017, 14:21
03.miscellaneous	26/06/2018, 13:58	19/09/2017, 14:47	26.6 MB	--	20/03/2018, 09:46	05/10/2017, 14:21
04.presentations	03/09/2018, 15:20	19/09/2017, 14:45	2.02 GB	--	23/11/2018, 15:35	05/10/2017, 14:21
05.databases_and_refseq	01/03/2018, 15:06	24/08/2017, 16:09	1.24 GB	--	26/02/2018, 10:27	05/10/2017, 14:21
06.raw_data_from_xp	27/11/2018, 09:35	07/08/2017, 14:49	133...GB	--	16/11/2018, 11:36	30/01/2018, 09:24
07.info	07/12/2018, 11:28	19/09/2017, 14:47	7.1 MB	--	31/05/2018, 14:28	05/10/2017, 14:21
08.internships	27/09/2017, 14:09	27/07/2017, 17:24	257 MB	--	19/02/2018, 16:53	05/10/2017, 14:21
09.protocols	19/06/2018, 10:04	24/08/2017, 16:12	819.4 MB	--	30/08/2018, 15:34	05/10/2017, 14:21
10.EPS	27/09/2017, 14:10	19/09/2017, 14:47	37.8 MB	--	--	05/10/2017, 14:21
11.results	24/08/2018, 10:25	19/09/2017, 14:49	8.03 GB	--	16/11/2018, 14:48	30/01/2018, 09:24
12.outreach	12/07/2018, 11:19	24/08/2017, 16:12	44.4 MB	--	--	05/10/2017, 14:21
13.teaching	19/09/2017, 14:48	19/09/2017, 14:48	58.8 MB	--	--	05/10/2017, 14:21
14.courses	08/10/2018, 15:55	24/08/2017, 16:12	905.9 MB	--	01/11/2018, 10:32	09/11/2017, 14:40
15.germplasm	12/07/2018, 11:19	30/01/2018, 09:25	98 KB	--	--	30/01/2018, 09:25
16.paper_reviews	13/04/2018, 10:16	11/04/2018, 16:15	2.4 MB	--	11/04/2018, 16:47	11/04/2018, 16:15
bleekerlab.github.io	01/02/2018, 16:27	30/01/2018, 09:43	13.5 MB	--	--	30/01/2018, 09:43
README.txt	Today, 14:53	26/01/2018, 17:14	486 bytes	--	Today, 14:52	30/01/2018, 09:25

Outline: the research data life cycle



2. Tools (R,Python,Shell)

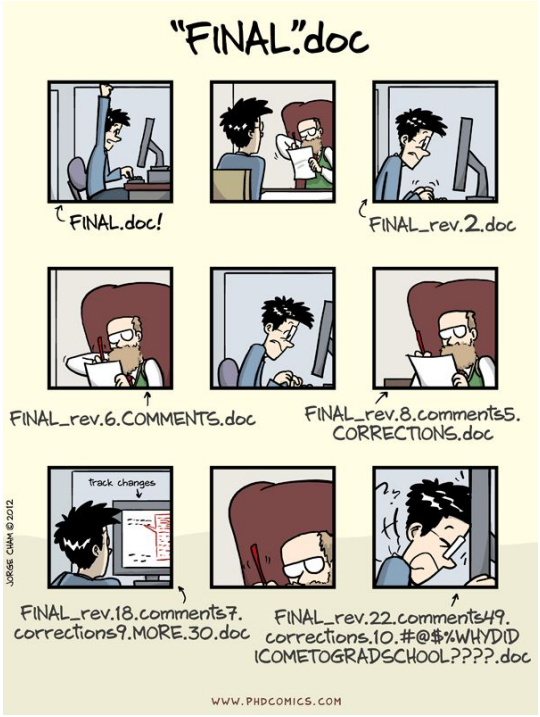
3. Project management

4. Good practices

4. Good practices in programming

Photo by Glenn Carstens-Peters on Unsplash

Why are good practices needed?



Good practices in programming

1. **Literate programming**
    a. Document your code
    b. Give functions and variables meaningful names
2. **Re-use code**
    a. Decompose programs into functions
    b. Eliminate duplications (make functions)
    c. Use well-maintained libraries
3. **Dependencies**: explicit what your program depends on
    a. Libraries
    b. Software versions
    c. Operating System (OS)
    d. Use containers and virtual environments (Docker,conda)
4. **Version control:** control and see what has changed

Wilson G, Bryan J, Cranston K, Kitzes J, Nederbragt L, et al. (2017) Good enough practices in scientific computing. PLOS Computational Biology 13(6): e1005510. https://doi.org/10.1371/journal.pcbi.1005510

Good practices in programming

1. **Literate programming**

```
# function to count from 1 to 5
def my_counting_function():
        """This function counts from 1 to 5 """
        for i in range(1,6):
            print(str(i))
```

2. **Re-use code**

```
# import the pandas data science library
import pandas as pd
```

3. **Dependencies**: explicit what your program depends on

```
# use conda
conda env create --name myenvironment
```

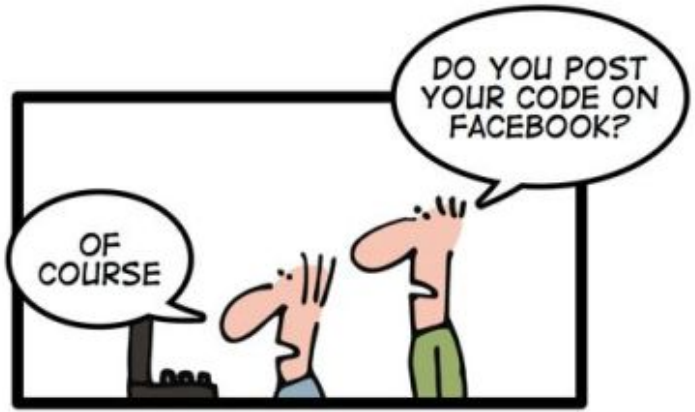
4. **Version control:** control and see what has changed

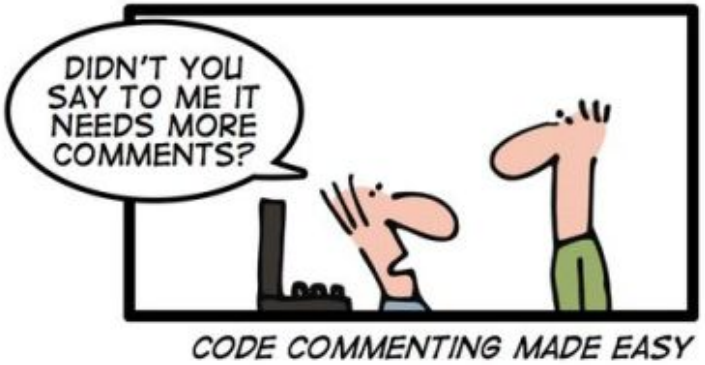
```
# stage your changes, commit and push
git add myscript.py
git commit -m "fixed mistake in
my_counting_function" myscript.py
```

Wilson G, Bryan J, Cranston K, Kitzes J, Nederbragt L, et al. (2017) Good enough practices in scientific computing. PLOS Computational Biology 13(6): e1005510. https://doi.org/10.1371/journal.pcbi.1005510

Literate programming

Code documentation





Literate programming

Code documentation

“Code is more often read than written.” *Guido Van Rossum*

You’re writing code documentation because:

- You will use your code in 6 months
- You want others to use your code and cite you
- You want to encourage others to contribute to your code
- You foster open science, reproducibility and good science ;-)

“It doesn’t matter how good your software is, because **if the documentation is not good enough, people will not use it.**“ *Daniele Procida*

Use of notebooks for reporting (Jupyter notebooks, Rmarkdown)

Literate programming

Code documentation

“Code is more often read than written.” *Guido Van Rossum*

```
def say_hello(name):
    """A simple function that says hello... Richie style"""
    print(f"Hello {name}, is it me you're looking for?")
```

```
>>> help(say_hello)
Help on function say_hello in module __main__:

say_hello(name)
    A simple function that says hello... Richie style
```

Re-use code

If you copy and paste several times the same code, make a function!

Easier to maintain, test and debug a few lines in one place

Import libraries and packages: `import pandas as pd`

Often the best packages and libraries have a thorough documentation.

Import your own functions:

In Python, if you have a “helper_functions.py” file in the same directory:

```
from helper_functions import my_custom_function
```

In R, if you have a “helper_functions.py” file in the same directory:

```
source("helper_functions.R")
```

Code dependencies

Your code depends on others' code.
List the dependencies: libraries, modules, etc.

In R, write the imports at the beginning of your script and get the session info at the end

```r
library("tidyverse")
sessionInfo()
```

In Python, import everything at the beginning. Make virtual environments

```python
import pandas as pd
```

Handling dependencies with containers

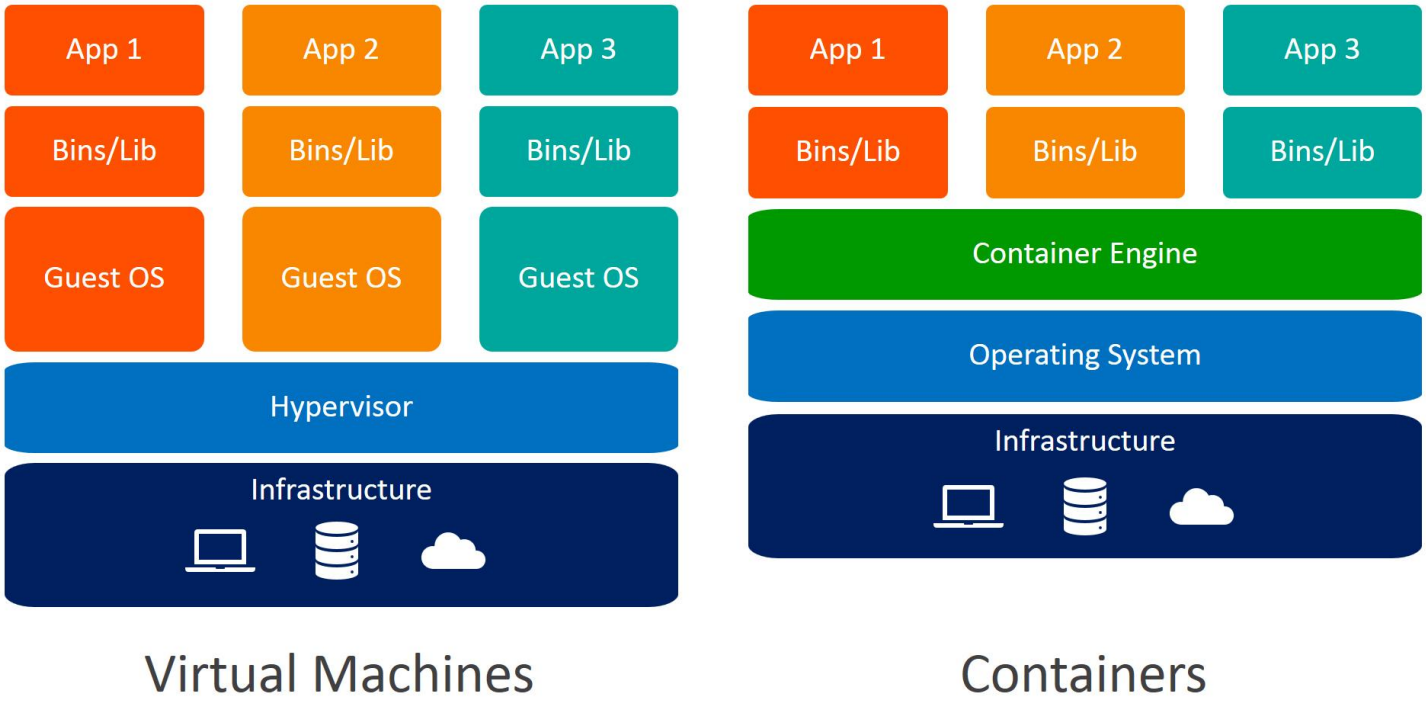
In bioinformatics, one single analytical pipeline will rely on dozens of different softwares (versions)

Containerization, allows one to package one or more programs together with everything that is needed to run those programs.

Singularity

docker

Virtual Machines & Containerization



Virtual Machines

Containers

Singularity images

Singularity has many interesting features for bioinformatics

1. Does not need root privilege to run (lab supercomputer or HPC environment) (not like Docker)
2. Singularity images can be built easily on a laptop (Singularity is an actual file)
3. Images can be shared (Singularity Hub) from a Github repository. Just a text file.

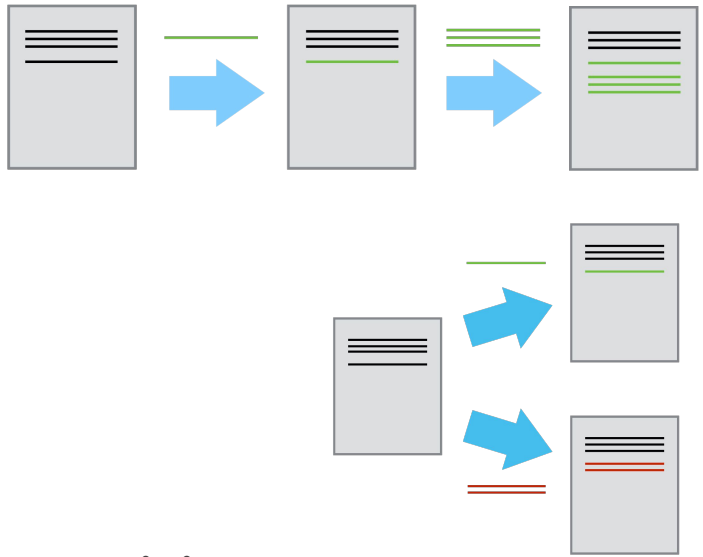
```
BootStrap: docker
From:dbgannon/ubuntuplus

%files

%labels
MAINTAINER gannon

%environment
LD_LIBRARY_PATH=/usr/local/lib
export LD_LIBRARY_PATH=/usr/local/lib

%runscript
   echo "This is what happens when you run the container..."
   /usr/bin/ring-simple
```

Using a version control system

Saves every change you've made

Allows real-time collaboration (not "wait for that email")

You know who has done what and when in a document



Version control is the lab notebook of the digital world!

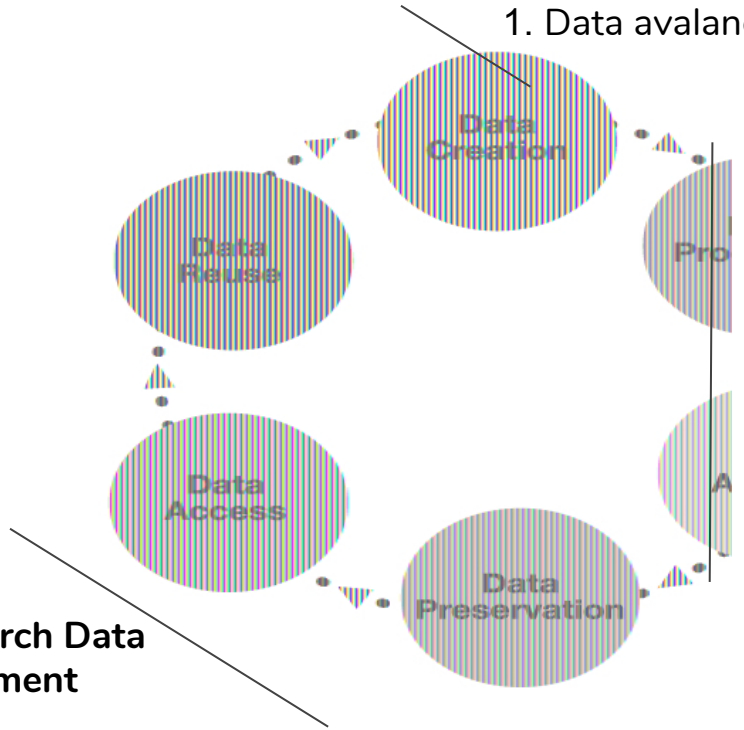
Using a version control system

Demo time using Github!

- Go to Github.com and make an account
- Create a repository
- Write a protocol “protocol.md” online
- Make some changes
- Compare the changes

Outline: the research data life cycle

1. Data avalanche



2. Tools (R,Python,Shell)

3. Project management

4. Good practices

5. Research Data Management

5. Research Data Management

Photo by Samuel Zeller on Unsplash

5. Research Data Management

Photo by Max Langelott on Unsplash

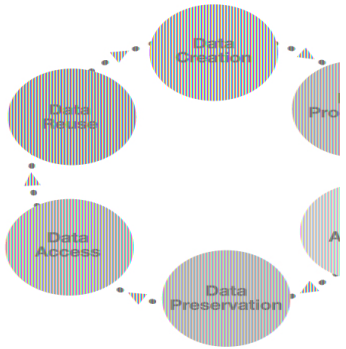
Some definitions

What are data? Every information gathered necessary to reproduce a scientific result:

- numbers in spreadsheet, pictures, lab notebooks, etc.

R**esearch Data management** occurs at every step of the research data life

- before / during / after your project

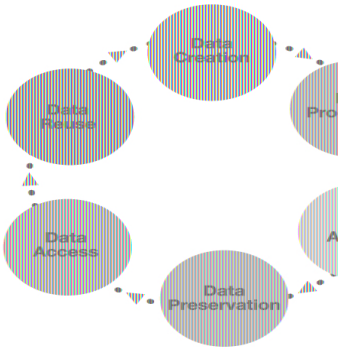


Data management is also called “**Data Stewardship”**

Why research data management?

Good research data management practice allows:

- ensure that data will be properly saved
- reliable verification of results (reproducibility / research integrity)
- allows new innovative research built on existing information.
- helps transdisciplinary investigations



Alisa Surkis and Kevin Read (2015). Research data management J Med Libr Assoc. 103(3):154–156. doi: 10.3163/1536-5050.103.3.011

Overview

Data Management Plans

Active Data Management

Data Backup

Data Sharing

Data Preservation and Archiving

Data Security

http://researchdata.berkeley.edu/

5. Data Management Plan

Data Management Plans

Dutch TechCenter for Life Sciences: https://ds-wizard.org/

5. Data Management Plan

Why a DMP? To ensure that you know how research data will be treated during and after your research project.

Ensures that data will be correctly treated: data format, backup, long-term storage, etc.

Explicit data access: licenses, public repositories (e.g. NCBI), etc.

Specify the associated costs….and who will pay for it!

5. Data Management Plan

What does a DMP should contain?

- Information about the data and their experimental information (metadata)
  - data description and format
  - metadata format and standards
  - data storage and backup
  - data organisation: naming conventions, version control
  - how will data be processed? Which softwares will be used?
- People responsible for the data management
- Access and intellectual property
- Longevity: how many years should the data be maintained?
- Budget, funding...

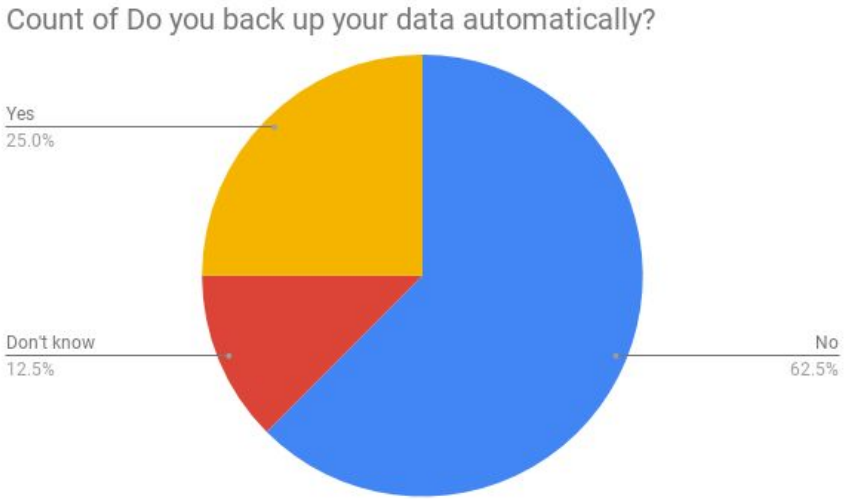
https://en.wikipedia.org/wiki/Data_management_plan

5. The FAIRness of data

The FAIR principles:

- Findability: data + code have a persistent identifier & rich metadata
- Accessibility: open protocol, accessible metadata
- Interoperability: controlled vocabulary, readable by machine & humans alike
- Reusability: domain-specific community standards, license, etc.

5. Data backup



Is your data automatically backed up?

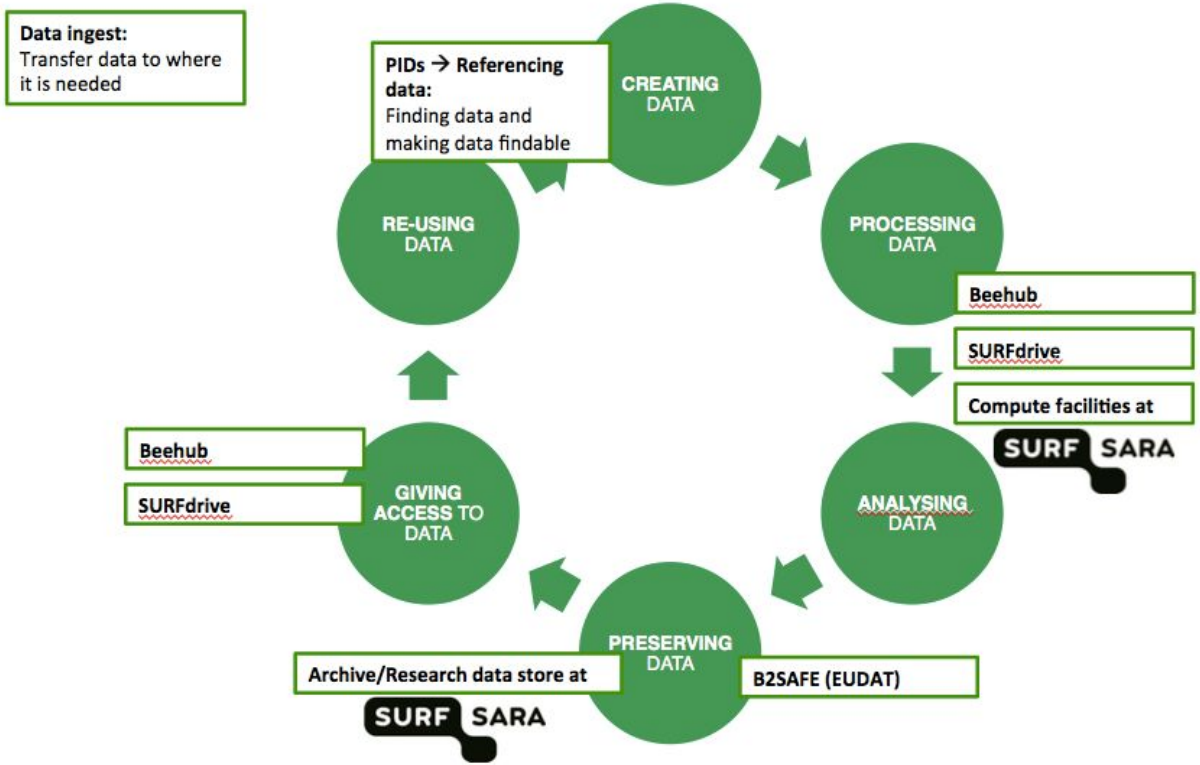
You want something:

- automatic (don't count on yourself!)
- cheap
- reliable

Often commercial solutions: Google, JottaCloud, etc.

SURF is the collaborative ICT (Information and Communication Technology) organisation for Dutch education and research.

5. SURF Services



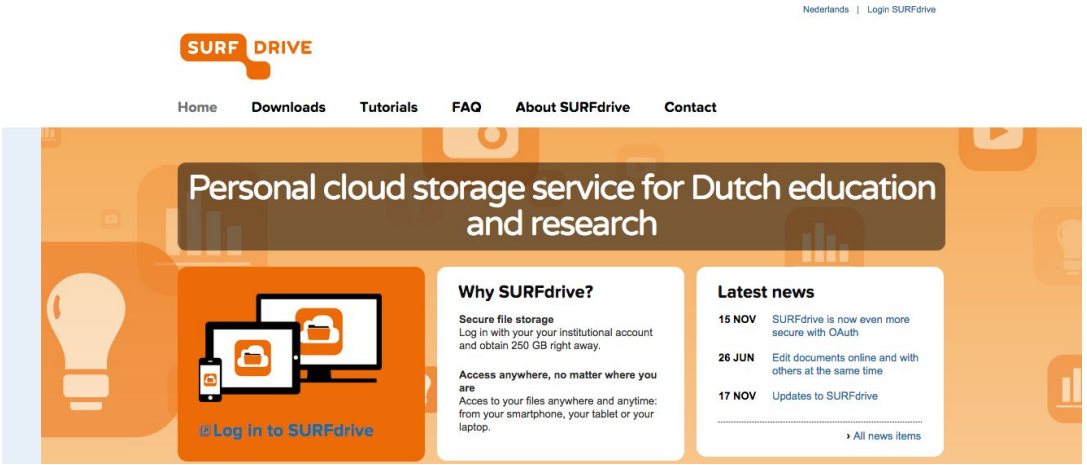
5. Data sharing / SURFdrive

SURFdrive can be used to share data within a research group

An institutional, secure, national “Dropbox/Google Drive” solution.

SURFdrive complies with all Dutch and European privacy legislation. The data are stored safely in the Netherlands.

250GB available for free.



5. Data sharing / SURF file sender

SURF FILESENDER

Home About SURFfilesender FAQ Contact

Send large files securely and encrypted

Start transfer

Why choose SURFfilesender?

- Send files of up to 500 GB
- Free for you as a user
- Sent via Dutch servers
- End-to-end encryption (up to 2 GB)

More information about SURFfilesender

Let's get started!

SURFfilesender is very easy to use:

Step 1: click "Start transfer" and log in using your institution details
Step 2: enter the recipient and select your file
Step 3: click "Send"

And you're done!

See also the frequently asked questions

Up to 500Gb!

Dutch servers

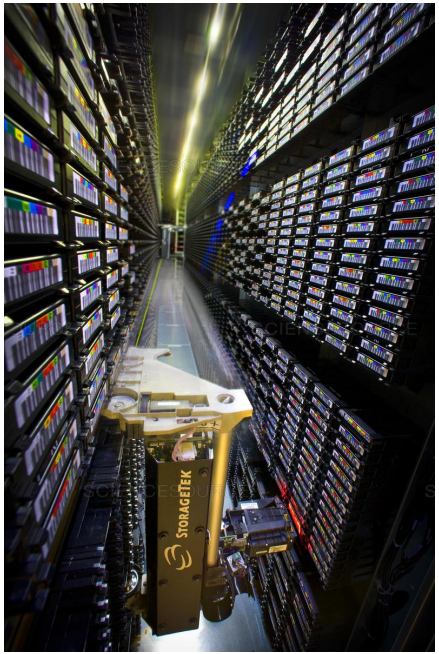
Data Preservation and Archiving

5. Data archiving (frozen)

Tape Archive

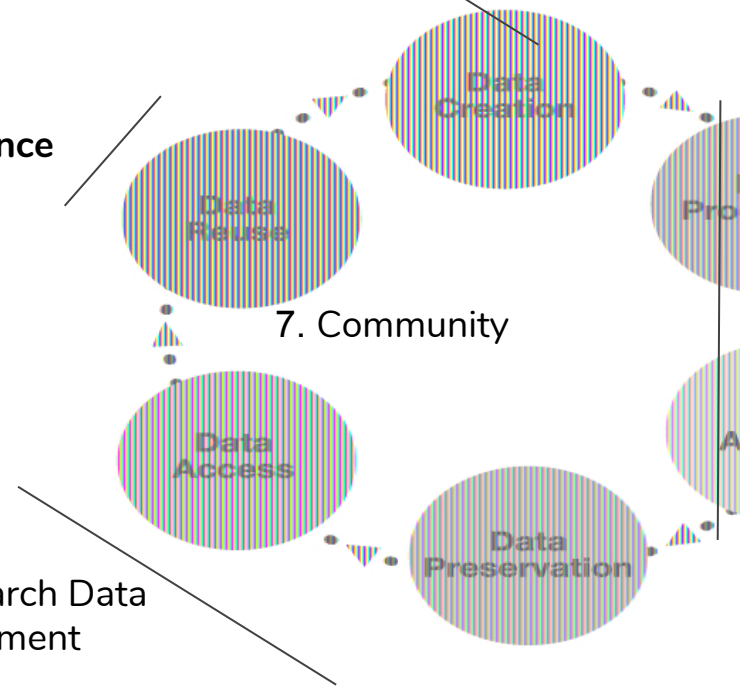
SURFsara Archive service

Zenodo (up to 50Gb per dataset)



Outline: the research data life cycle

1. Data avalanche



6. Open Science

7. Community

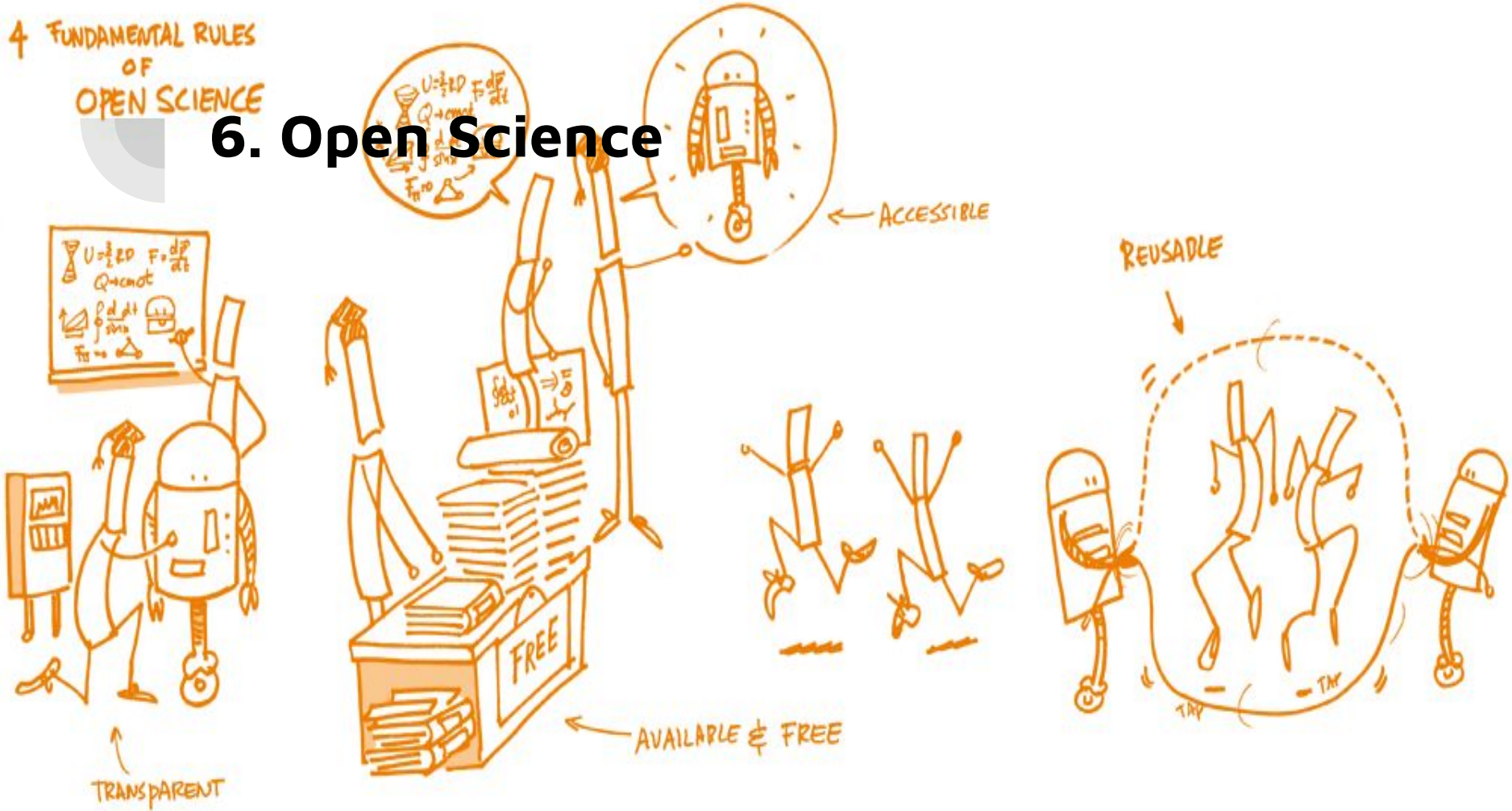
2. Tools (R,Python,Shell)

3. Project management

4. Good practices

5. Research Data Management

6. Open Science



Open Science definition

Open Science is the practice of science in such a way...

...that others can collaborate and contribute...

...where research data, lab notes and other research processes are freely available...

...under terms that enable reuse, redistribution and reproduction of the research and its underlying data and methods.

A political statement + a daily practice + a change of mind for scientists

Open Science: what's the urge?

Plan S: every publicly-funded research in the EU is to be made open access in journals in 2020

Research data/code from publicly-funded research should be freely available

Foster collaboration between labs and disciplines through easier exchange of data.

The "reproducibility crisis": hard to reproduce others' work, "null results", confirmation of others.

Improve research integrity: avoid data manipulation. Cases with PubPeer & article retractation

Open Science: what can you do?

Publish datasets before/along your biological story for instance in *Nature Scientific Data*

Candela et al. (2015) Data journals: a survey.https://doi.org/10.1002/asi.23358

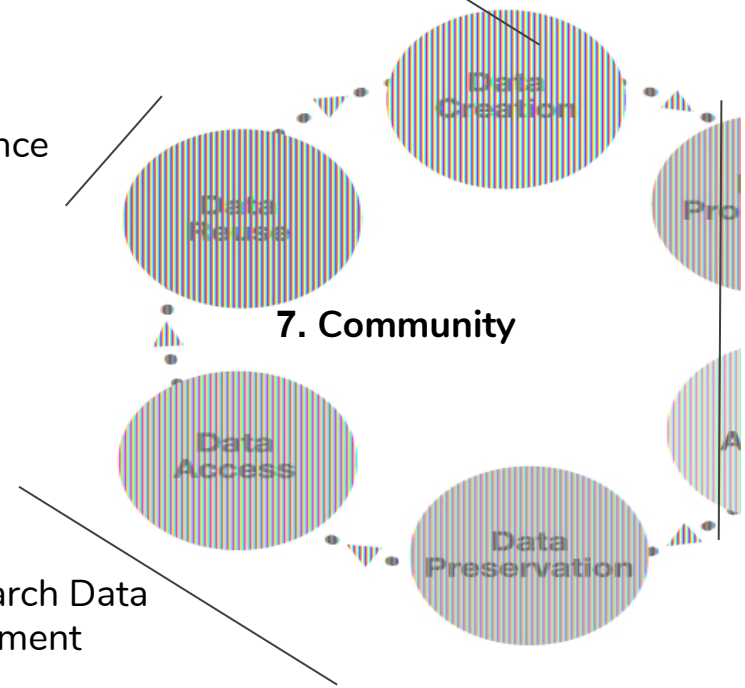
Deposit data in public repositories: Zenodo, FigShare, etc.

Treat your code openly: license, documentation, integration GIthub-Zenodo

Publish your software in a dedicated journal e.g. Journal of Open Software

Outline: the research data life cycle

1. Data avalanche



2. Tools (R,Python,Shell)

3. Project management

4. Good practices

5. Research Data Management

6. Open Science

7. Build a local community

Photo by Slava Bowman on Unsplash

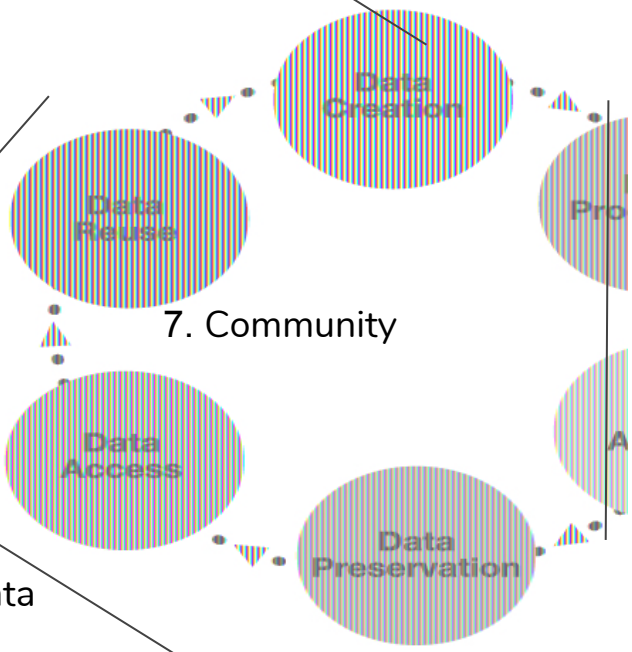
7. Build a local community

https://slides.com/mgalland/the-amsterdam-science-park-study-group-2

Photo by Slava Bowman on Unsplash

Conclusion: the research data life cycle

1. Data avalanche



6. Open Science

7. Community

2. Tools (R,Python,Shell)

3. Project management

4. Good practices

5. Research Data Management

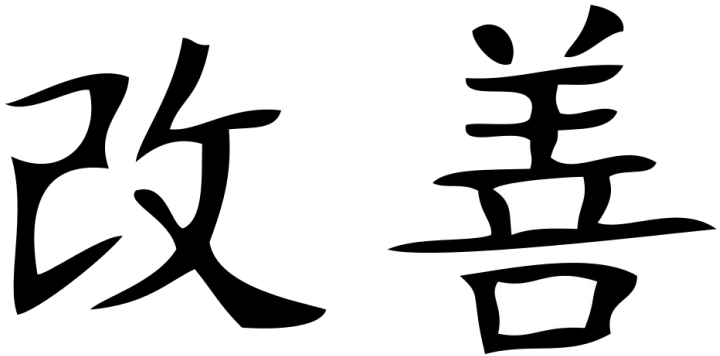
Conclusions

Documentation is key: data, scripts, project, workflows…

“Every scientist is a data scientist!”
-- Till Bey, 2018

“It’s never too late to start coding”
-- Sarah Stolle, 2018

Kaizen: continuous improvement



“change” “better”

EPS Get together 2019

"*Homo biologicus informaticus*":
must know & good practices for the future Life Scientist

Marc Galland - Data Scientist/Manager (University of Amsterdam) & Founder BioData Services
www.mgalland.info
www.biodataservices.eu

Where to find this presentation?

Archived on Zenodo: the archive of the CERN particle collider

Resources for Shell and R

- Software Carpentry Lesson on Shell: http://swcarpentry.github.io/shell-novice/
- The official R project website: https://www.r-project.org/
- The **Bioconductor project** (tools for genomics): https://www.bioconductor.org/
    - An example workflow: http://master.bioconductor.org/packages/release/workflows/vignettes/rnaseqGene/inst/doc/rnaseqGene.html
- **Hadley Wickham,** conceptor of ggplot2 and chief scientist at RStudio
    - Personal website with plenty of useful links: http://hadley.nz/
    - R for data science: https://r4ds.had.co.nz/

Resources for Python

- Python foundation: https://www.python.org/psf/
- Learning Python for non-programmers: https://wiki.python.org/moin/BeginnersGuide/NonProgrammers
- Ekmekci B, McAnany CE, Mura C (2016) An Introduction to Programming for Bioscientists: A Python-Based Primer. PLOS Computational Biology 12(6): e1004867. https://doi.org/10.1371/journal.pcbi.1004867
- Software Carpentry lessons on Python: https://swcarpentry.github.io/python-novice-inflammation/

Literature

Papers

- **Building a local community of practice in scientific programming for life scientists**. Stevens SLR, Kuzak M, Martinez C, Moser A, Bleeker P and Galland M (2018) PLOS Biology 16(11): e2005561.
- **Good enough practices in scientific computing**. Wilson G, Bryan J, Cranston K, Kitzes J, Nederbragt L, et al. (2017) Good enough practices in scientific computing. PLOS Computational Biology 13(6): e1005510.
- **Tidy data.** Hadley Wickham (2014). Journal of Statistical Software 59(10). A demo is also accessible here.
- **Computing Workflows for Biologists: A Roadmap.** Shade A, Teal TK (2015) Computing PLoS Biol 13(11): e1002303. doi:10.1371/journal.pbio.1002303.

Books

- **Vince Buffalo: "Bioinformatics Data Skills"**

Online resources

Online training: Data Camp, Coursera, etc.

Software Dependencies:

- https://cloud4scieng.org/singularity-a-container-system-for-hpc-applications/

Research Data Management (Data Stewardship)

- Presentation by Daphné van Beek (UMC) https://zenodo.org/record/2328660
- Data archives: Zenodo, FigShare, DataverseNL
- Code is also data: how to publish it using Github and Zenodo (UC Berkeley)

Forums: Biostars, Stack Overflow, R-bloggers, etc.

Tutorials & training materials

- Dave Tang: lots and lots of tips for computational biology and genomics
- GOBLET: training resource in bioinformatics (lots of material!)

Online resources

Open Science: https://open-science-training-handbook.gitbook.io/