Computational Reproducibility With Scientific Workflows: Analysing viral genomes with Nextflow

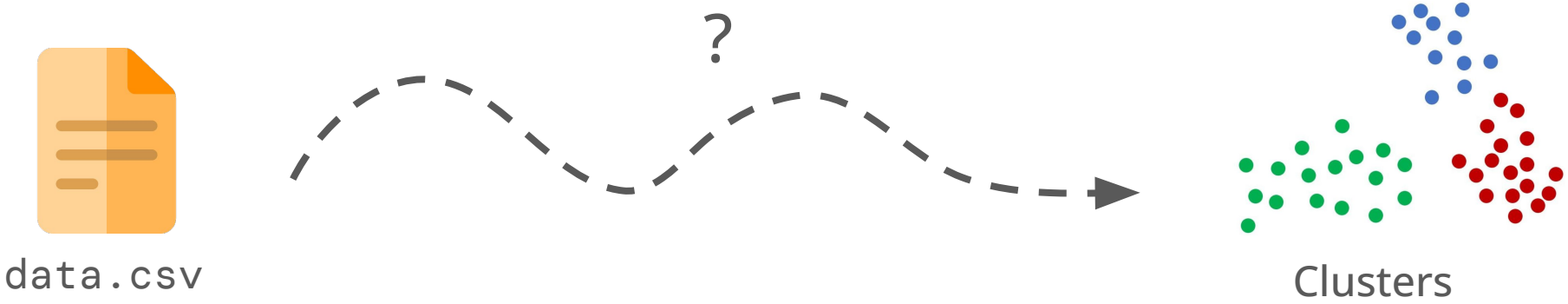
George Marchment, Sarah Cohen-Boulakia and Frédéric Lemoine

29/07/2025

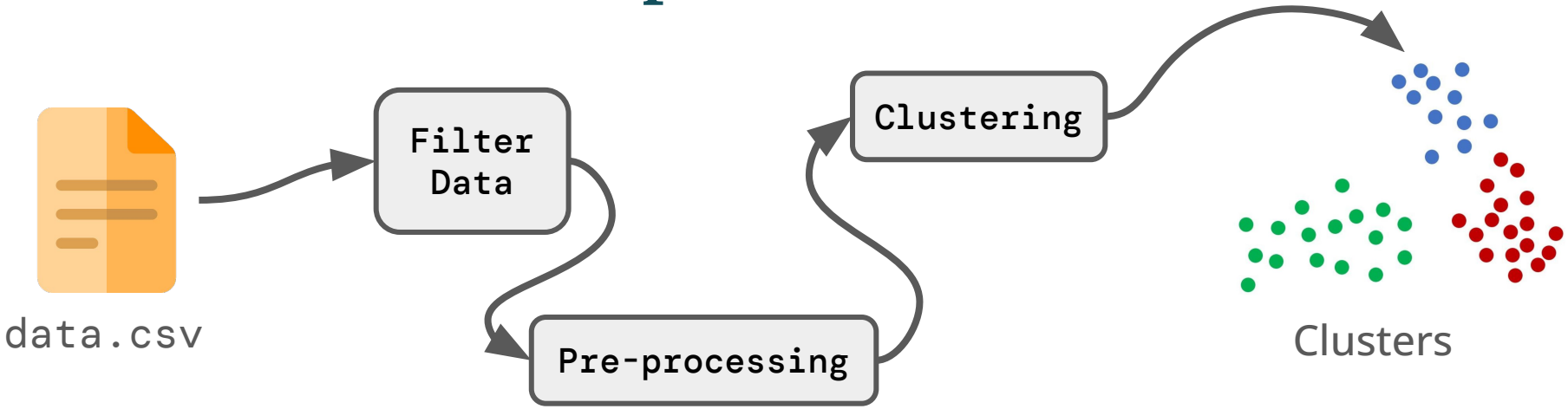
ACM REP '25

Motivational Example

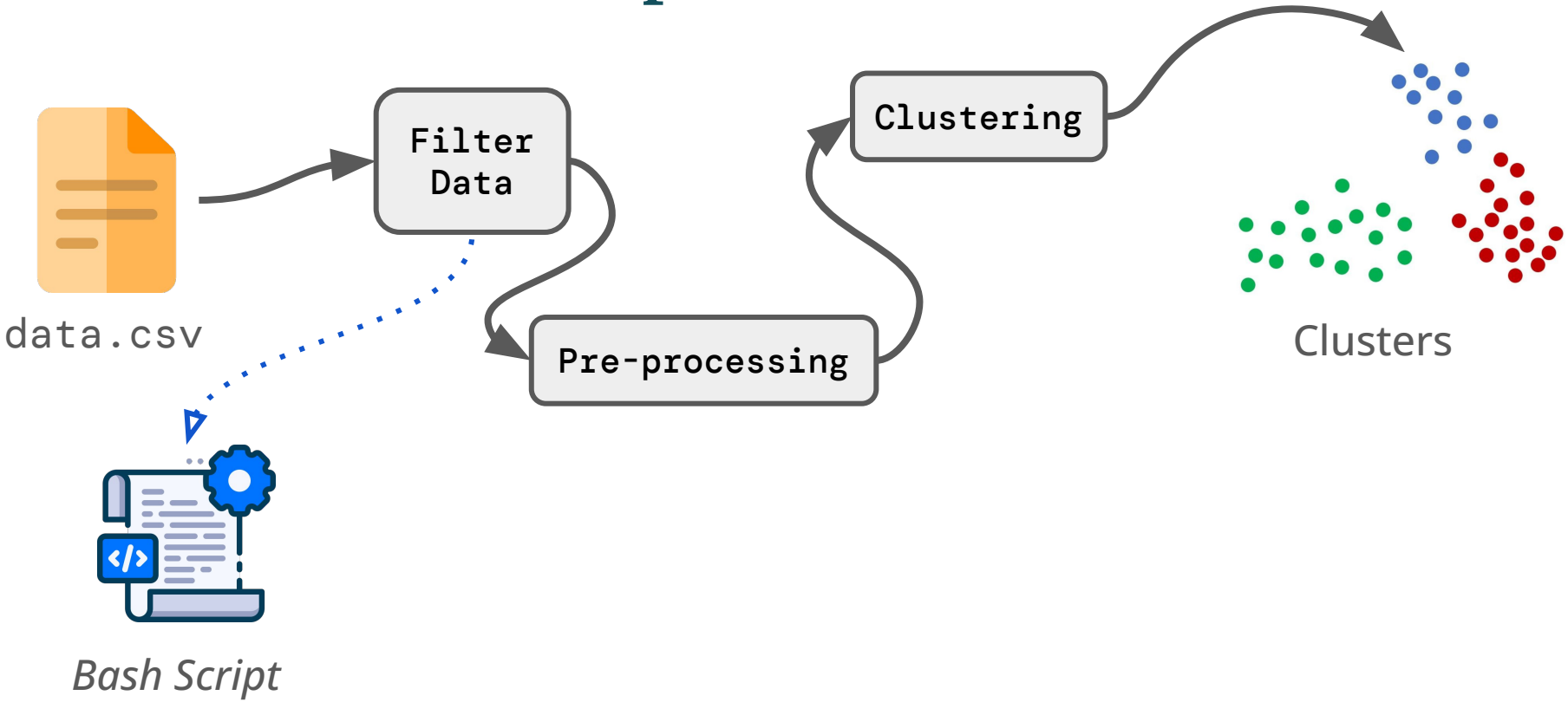
Motivational Example



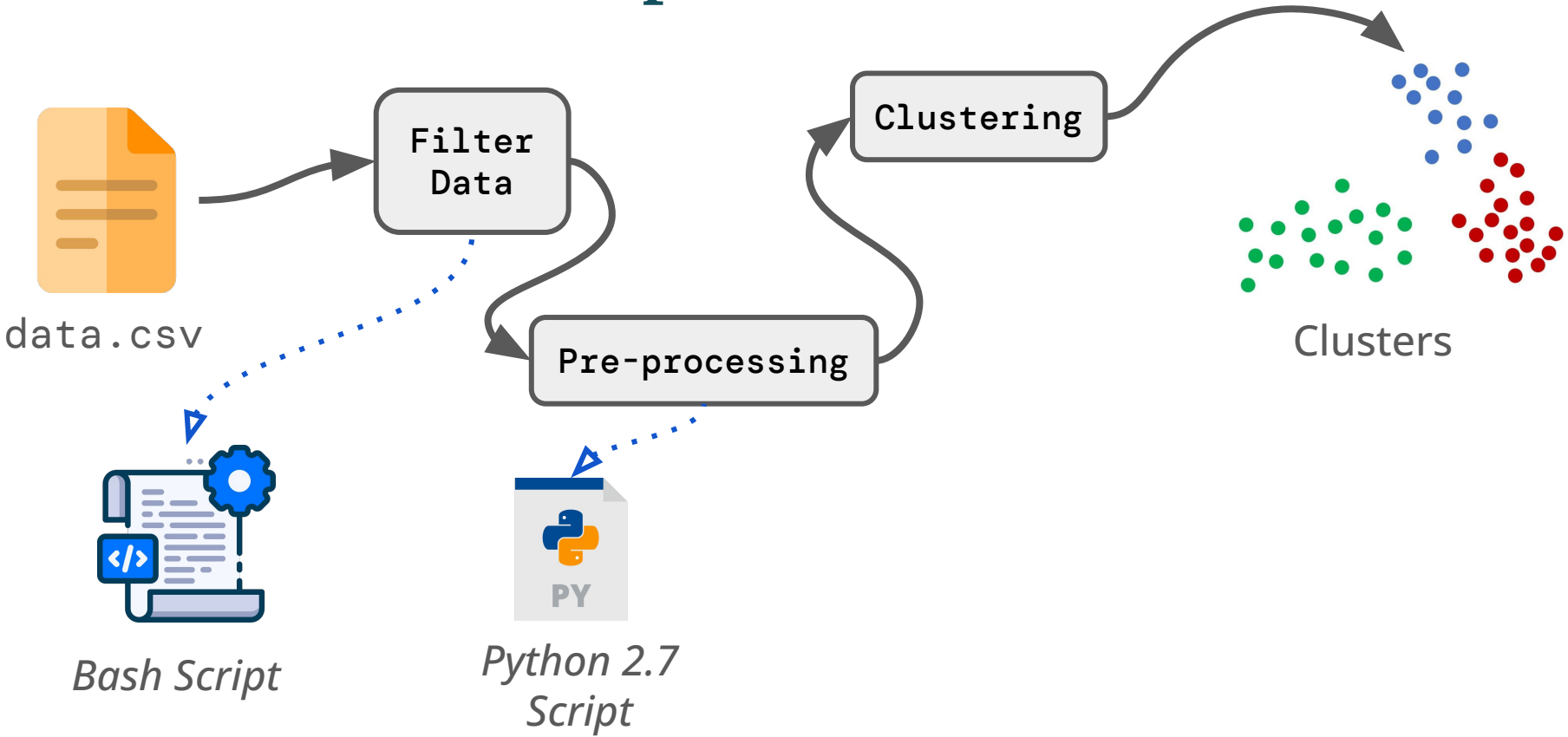
Motivational Example



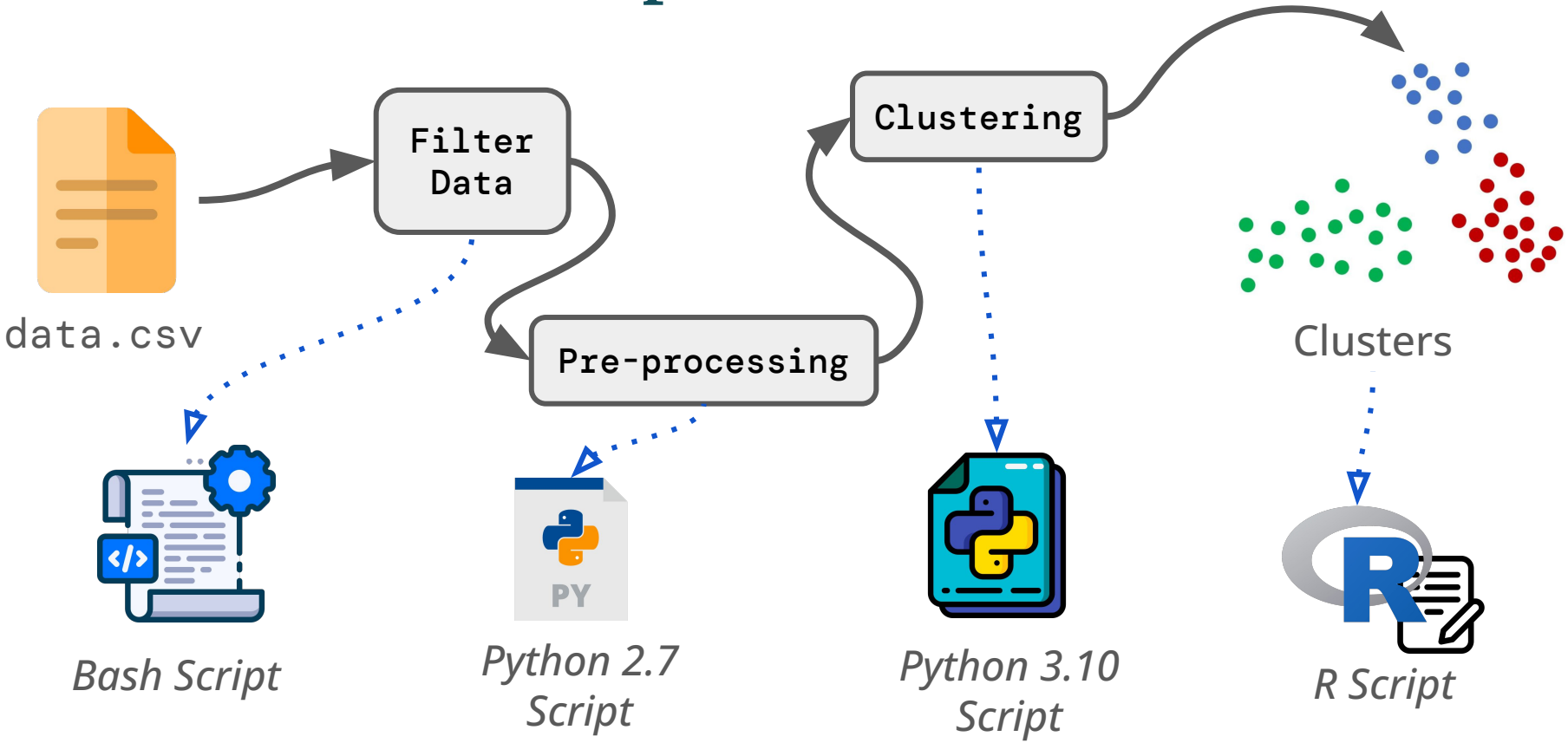
Motivational Example



Motivational Example

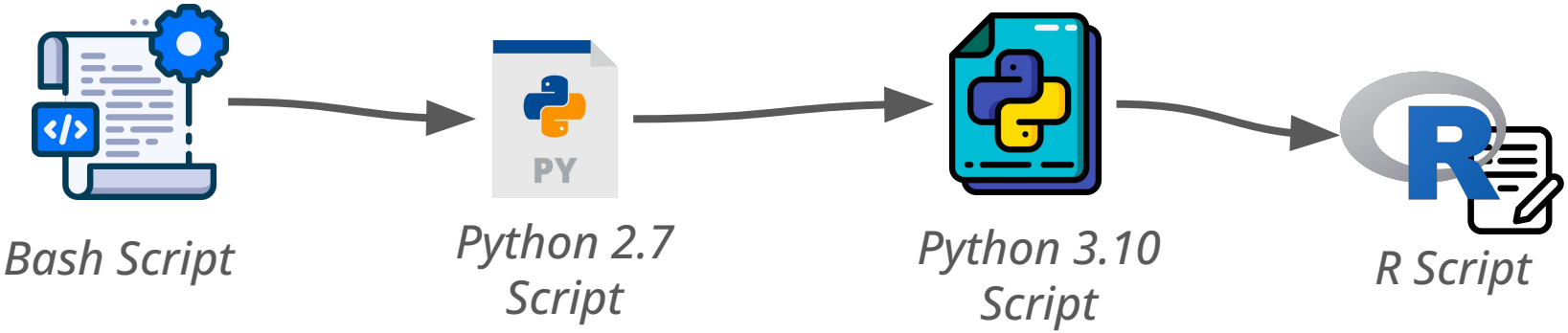


Motivational Example



Motivational Example

*How do we **orchestrate** all these different scripts together in a reproducible way?*

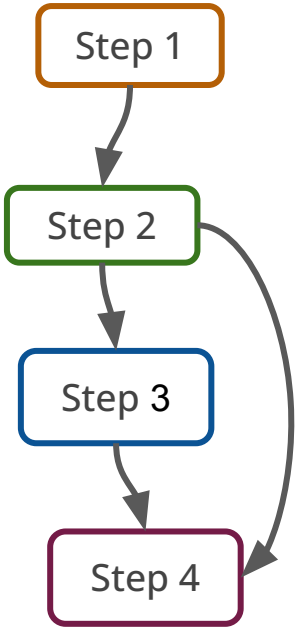


Objectives

- Acquiring basic **workflow concepts**
- Understanding the **capabilities of workflow management systems** in encapsulating heterogeneous code, scalability, software environment management, and computational resource management
- Learning how to **implement** simple workflows

1. Introduction to (Nextflow) Workflows (~1h)
    a. Motivational Example
    b. Objectives
    c. **Scientific Workflows**
    d. Software Containers
    e. Nextflow Workflows
    f. Presentation of the “Project”
2. Practical Work (~1h30)
3. Reproducibility Consensus (~30min)

Scientific Workflows



A **set of data analysis steps** that are **linked** together to form a more **complex** task

Workflow management system:

Apache Taverna

Galaxy PROJECT

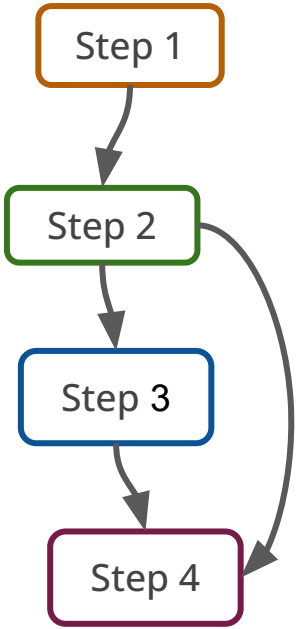
Drag and drop programming

nextflow

snakemake

Code based programming

Scientific Workflows



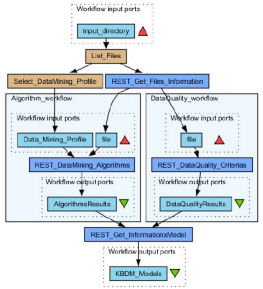
A **set of data analysis steps** that are **linked** together to form a more **complex** task

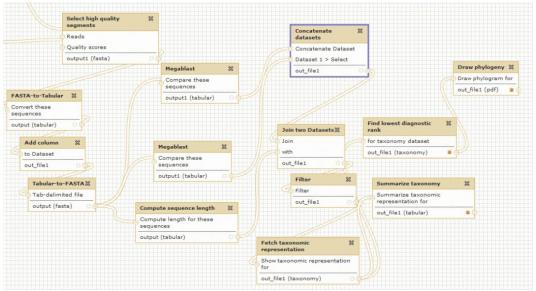
Workflow management system:

Apache Taverna
Galaxy PROJECT
Drag and drop programming

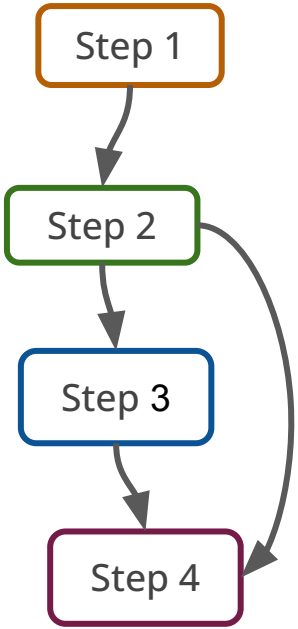
nextflow
snakemake
Code based programming

Development point of view:





Scientific Workflows



A **set of data analysis steps** that are **linked** together to form a more **complex** task

Workflow management system:

Apache Taverna

Galaxy PROJECT

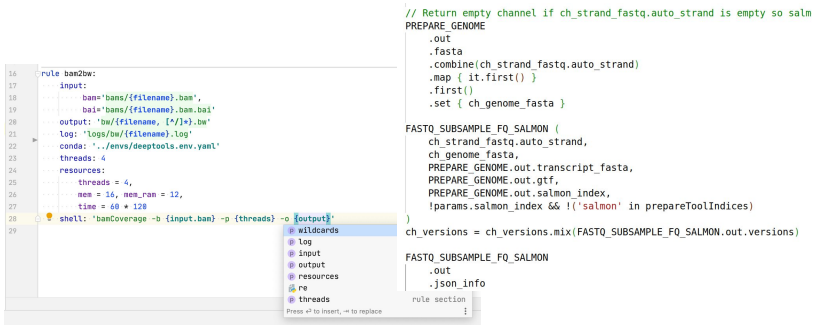
Drag and drop programming

nextflow

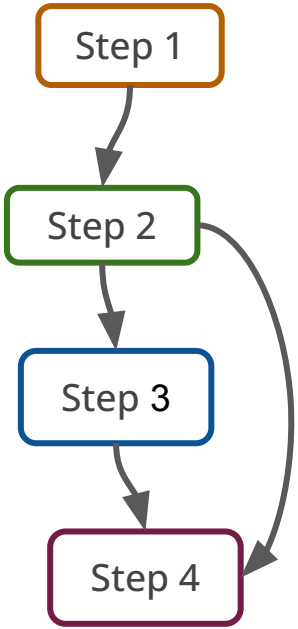
snakemake

Code based programming

Development point of view:



Scientific Workflows



A **set of data analysis steps** that are **linked** together to form a more **complex** task

Workflow management system:

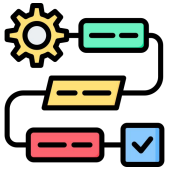
Apache Taverna
Galaxy PROJECT

Drag and drop programming

nextflow
snakemake

Code based programming

Share



Describe

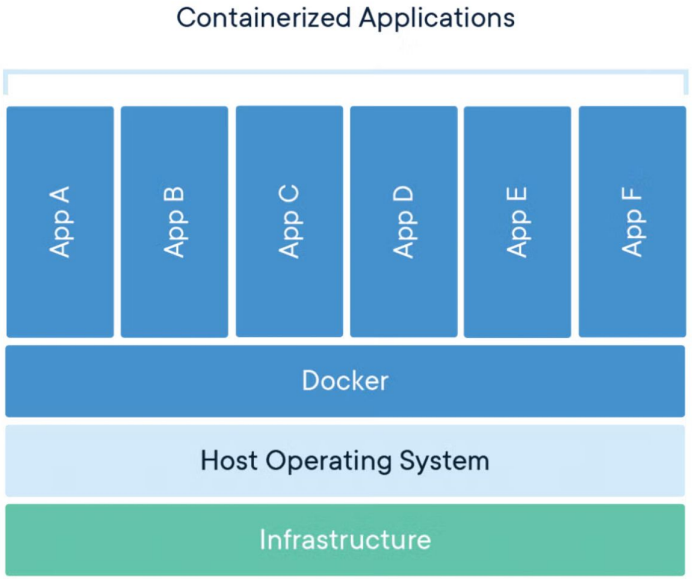
Execute

While guaranteeing a better **reproducibility** of analyses

1. Introduction to (Nextflow) Workflows (~1h)
    a. Motivational Example
    b. Objectives
    c. Scientific Workflows
    d. **Software Containers**
    e. Nextflow Workflows
    f. Presentation of the “Project”
2. Practical Work (~1h30)
3. Reproducibility Consensus (~30min)

Software Containers

- Are lightweight, standalone, and executable software packages that **include everything needed to run a piece of software**
- Are **lightweight**, **reproducible** and **consistent**
- Are totally **independent** to the host system and other software installed on the machine



docker.com

docker

Apptainer

1. Introduction to (Nextflow) Workflows (~1h)
    a. Motivational Example
    b. Objectives
    c. Scientific Workflows
    d. Software Containers
    e. **Nextflow Workflows**
    f. Presentation of the “Project”
2. Practical Work (~1h30)
3. Reproducibility Consensus (~30min)

Nextflow Workflows

- Are composed of **many simple tasks** which are chained together to form a more **complex pipeline**
- Can be composed of **steps written is mostly any programming language**:
  - Bash, Perl, Ruby, Python, etc...
- Are **flexible**, **scalable** and **reproducible**

```
// Script parameters
params.query = "/some/data/sample.fa"
params.db = "/some/path/pdb"

process blast_search {
  input:
  path query
  path db

  output:
  path "top_hits.txt"

  script:
  """
  blastp -db $db -query $query -outfmt 6 > blast_result
  cat blast_result | head -n 10 | cut -f 2 > top_hits.txt
  """
}

process extract_top_hits {
  input:
  path top_hits
  path db

  output:
  path "sequences.txt"

  script:
  """
  blastdbcmd -db $db -entry_batch $top_hits > sequences.txt
  """
}

workflow {
  def query_ch = channel.fromPath(params.query)
  blast_search(query_ch, params.db)
  extract_top_hits(blast_search.out, params.db).view()
}
```

Processes

- Are the **steps of the workflow**
- Are **user defined**
- Are **composed of different sections**

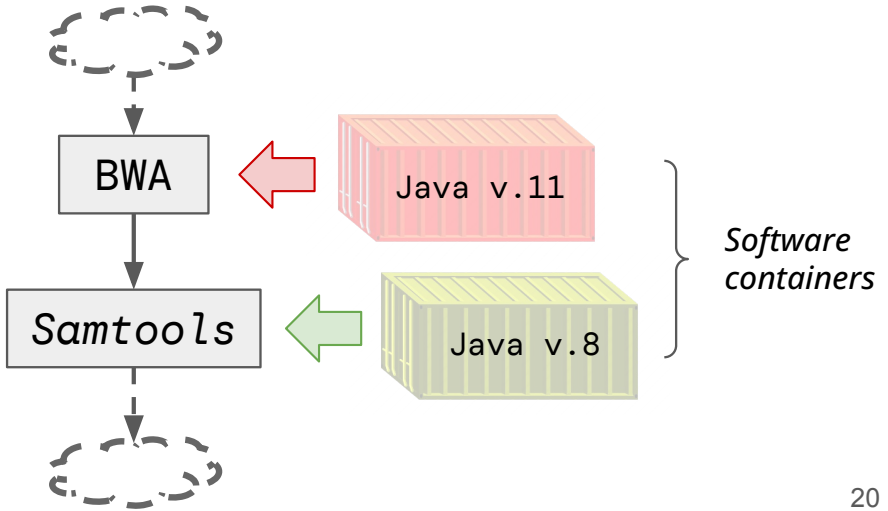
```
process example_process {
    label 'example'

    input:
        <input qualifier> <input name>

    output:
        <output qualifier> <output name> [, <option>: <option value>]

    '''
    echo "This is an example process!"
    '''
}
```

Processes

- Are the **steps of the workflow**
- Are **user defined**
- Are **composed of different sections**

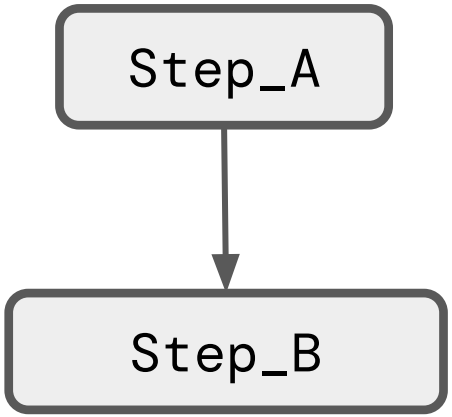
```
process example_process {
    label 'example'

    input:
        <input qualifier> <input name>

    output:
        <output qualifier> <output name> [, <option>: <option value>]

    '''
    echo "This is an example process!"
    '''
}
```

- Have **independent software environments** to one another (thanks to software containers)
- Are **chained together** to form the more complex pipeline



Channels and Operators

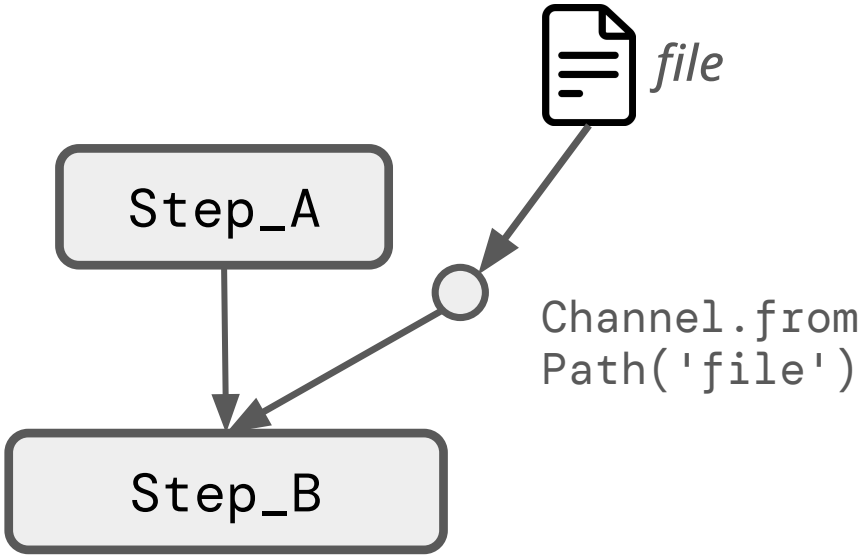
- Channels allow **to exchange data between the processes** of the workflow
- Channels can considered as **FIFO queues**



Channels and Operators

- Channels allow **to exchange data between the processes** of the workflow
- Channels can considered as **FIFO queues**

- Operators allows to **manipulate** channels, by
  - creating them, merging them, filtering them, etc...
- Operators are **pre-defined by Nextflow**
- There are **many different types of operators** (*today we are only gonna focus on the simplest ones*)



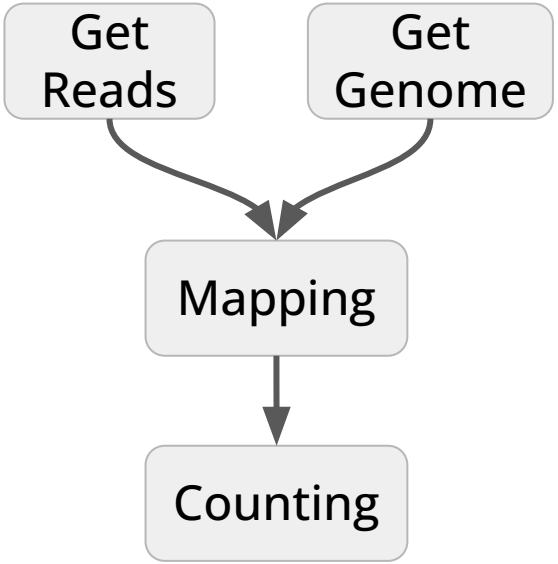
Workflow Main

- Is the **entry point of the workflow**
- It can be considered as the “main” as in generic programming (C, java, etc…)
- It is from this section **Processes can be called and Channels manipulated**

```
// Script parameters
params.query = "/some/data/sample.fa"
params.db = "/some/path/pdb"

process blast_search {
  input:
  path query
  path db

  output:
  path "top_hits.txt"

  script:
  """
  blastp -db $db -query $query -outfmt 6 > blast_result
  cat blast_result | head -n 10 | cut -f 2 > top_hits.txt
  """
}

process extract_top_hits {
  input:
  path top_hits
  path db

  output:
  path "sequences.txt"

  script:
  """
  blastdbcmd -db $db -entry_batch $top_hits > sequences.txt
  """
}

workflow {
  def query_ch = channel.fromPath(params.query)
  blast_search(query_ch, params.db)
  extract_top_hits(blast_search.out, params.db).view()
}
```

Configuration File

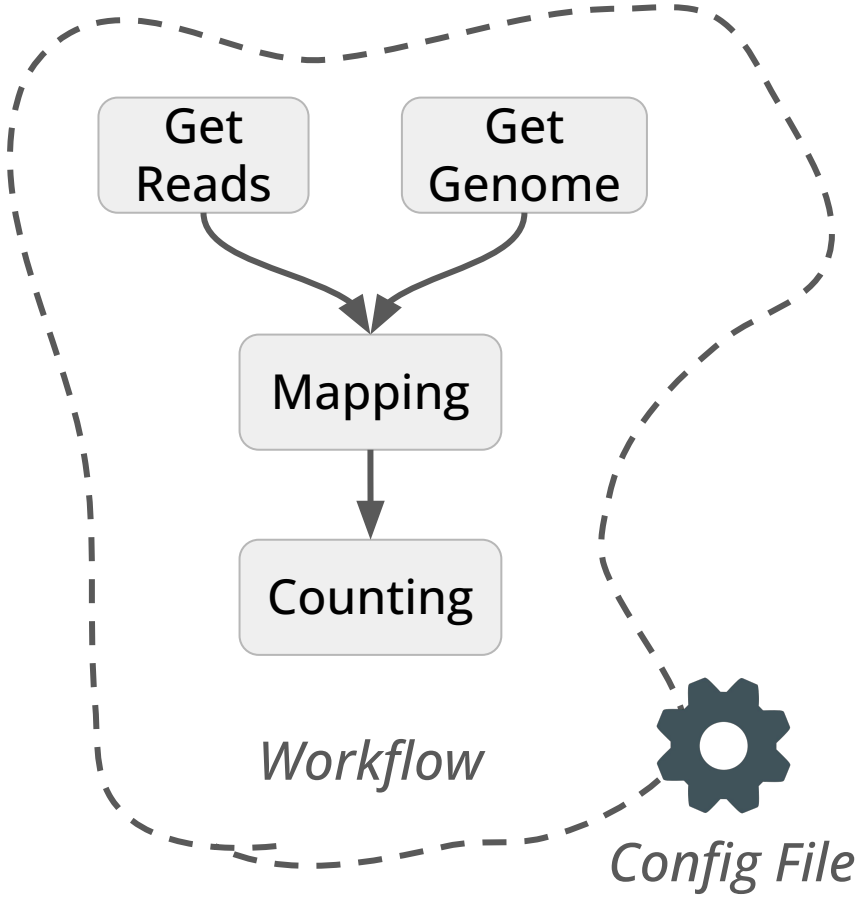
- Is **necessary to execute the workflow**



Workflow

Configuration File

- Is **necessary to execute the workflow**
- Defines the **configuration properties** of the workflow
- Is used to:
  - define which **executor to use**
  - the **process's environment variables**
  - **pipeline parameters**
  - etc...



Summary

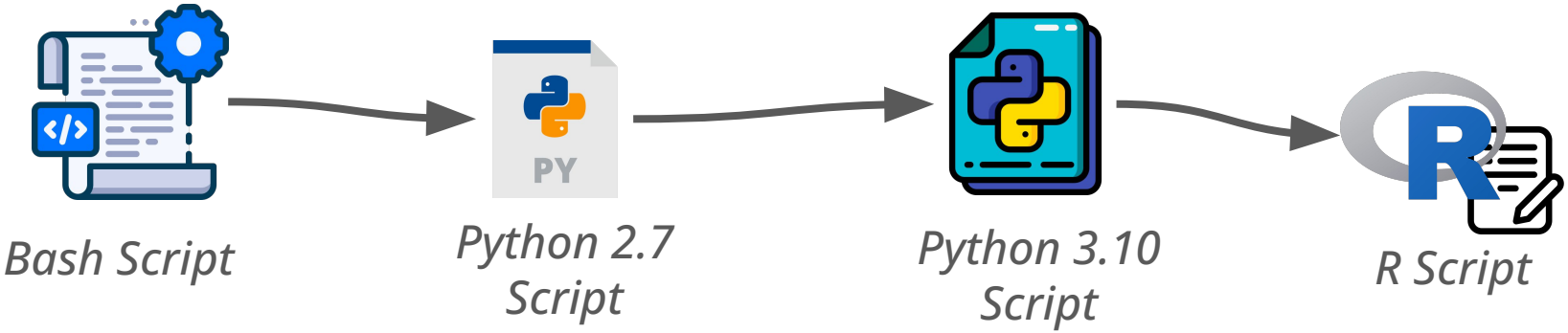
- Nextflow allow to develop **flexible**, **scalable** and **reproducible** workflows

- Nextflow workflows are composed of:
  - **Processes**
  - **Channels** (which can be manipulated thanks to **Operators**)
  - **Workflow Main**
  - **Configuration File**

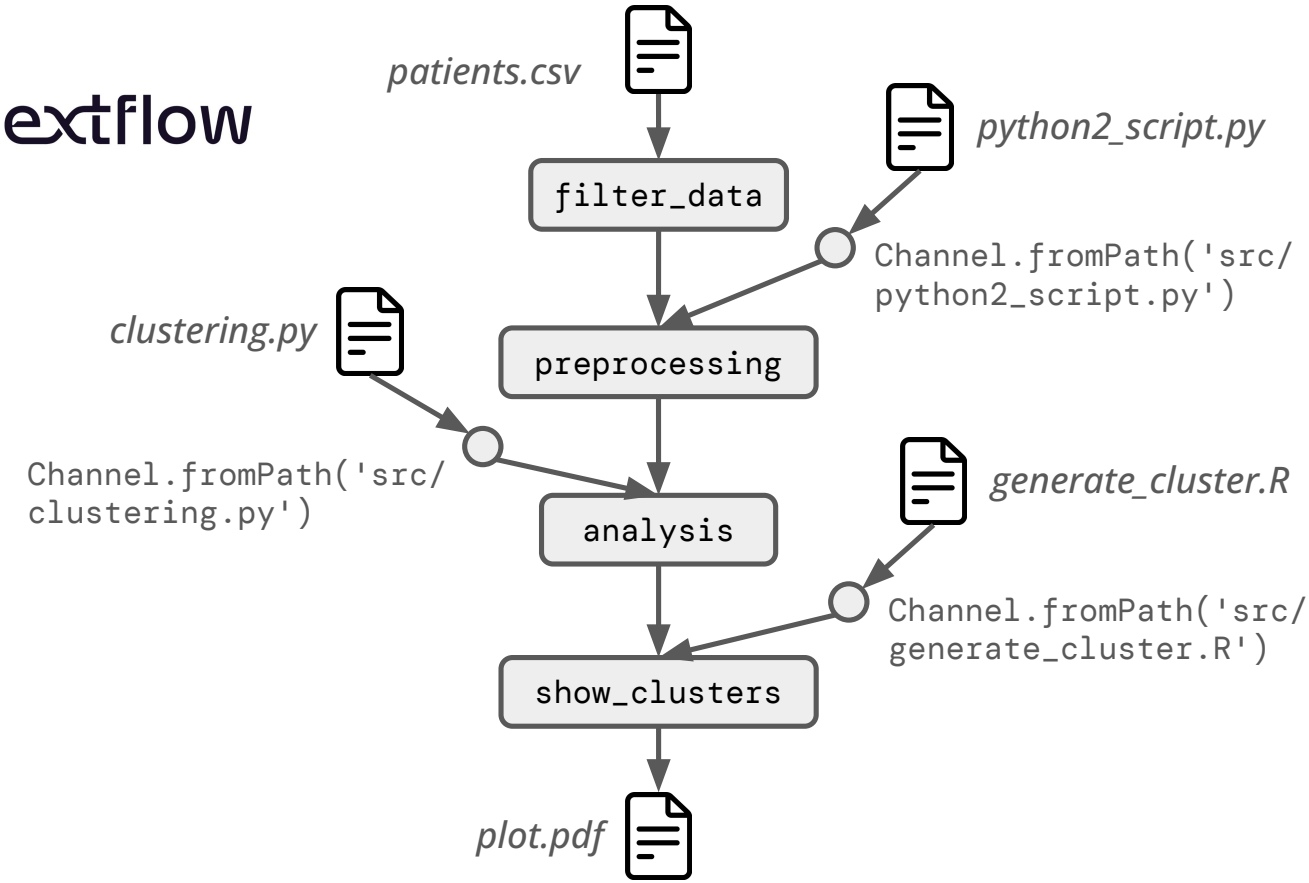
- For more information regarding Nextflow workflows, check out its documentation and tutorial https://www.nextflow.io/

Returning to the Motivational Example

*How do we **orchestrate** all these different scripts together in a reproducible way?*



Returning to the Motivational Example



Returning to the Motivational Example

- Link to workflow implementation:
  - https://github.com/George-Marchment/acmrep25/tree/main/example_use_case

- Command to run the workflow:

```
nextflow workflow.nf
```

Returning to the Motivational Example

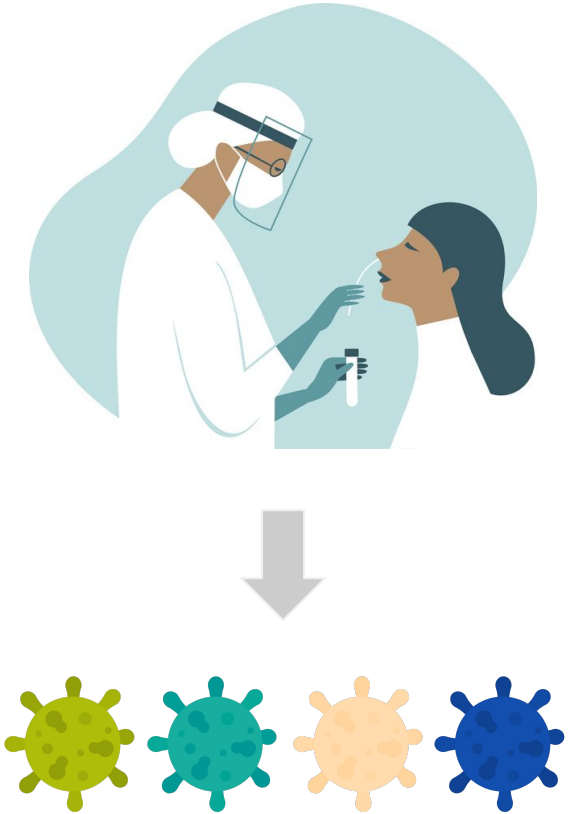
- Link to workflow implementation:
  - https://github.com/George-Marchment/acmrep25/tree/main/example_use_case

- Command to run the workflow:

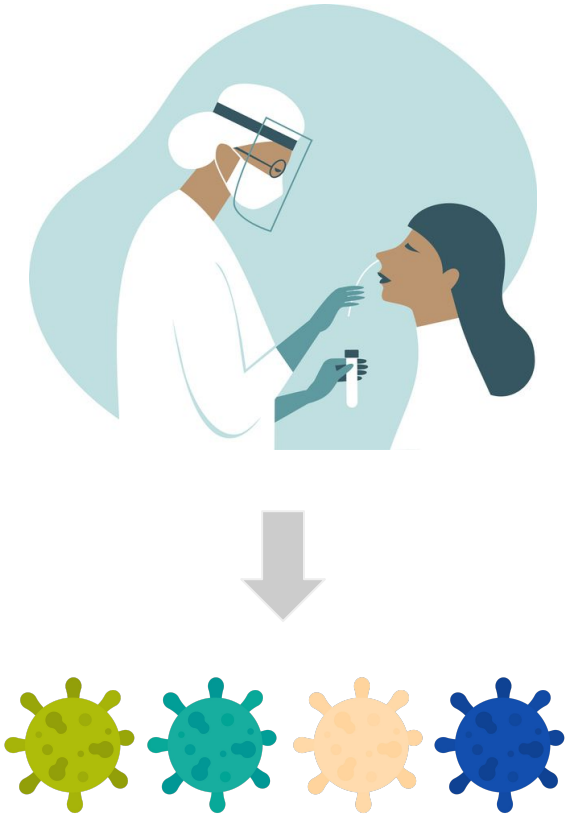
```
nextflow workflow.nf
```

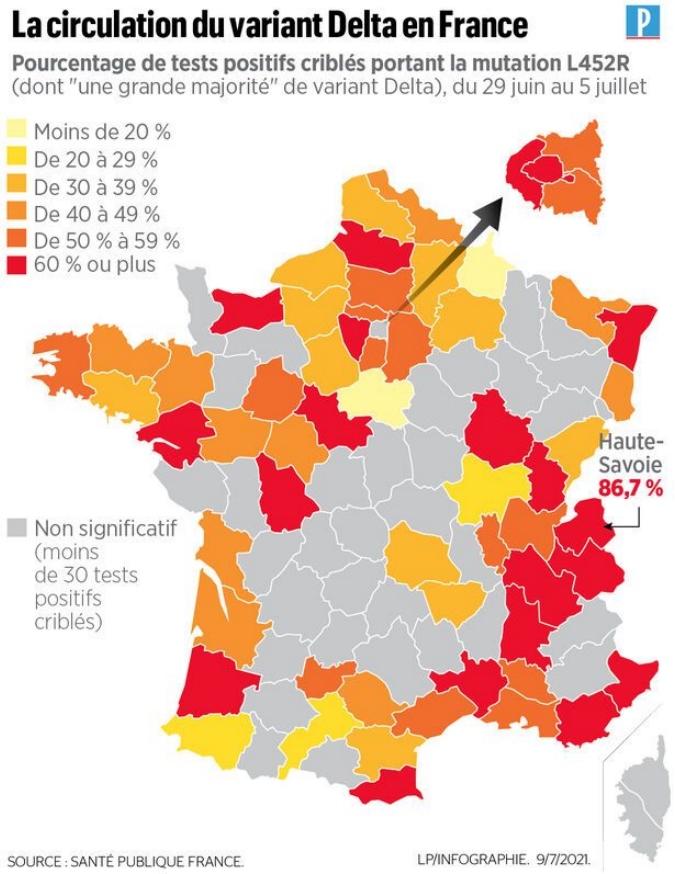
Demo

1. Introduction to (Nextflow) Workflows (~1h)
    a. Motivational Example
    b. Objectives
    c. Scientific Workflows
    d. Software Containers
    e. Nextflow Workflows
    f. **Presentation of the “Project”**
2. Practical Work (~1h30)
3. Reproducibility Consensus (~30min)



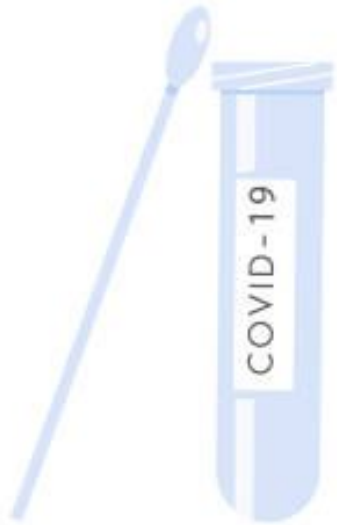
Identification of the **variant** in a person infected with the virus



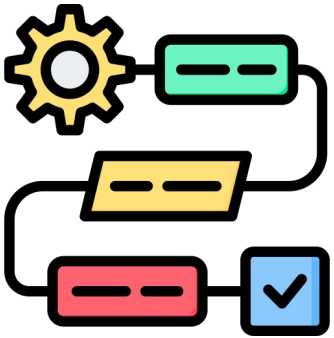


https://www.leparisien.fr/societe/sante/covid-19-et-variant-delta-en-france-ces-departements-ou-lepidemie-repart-09-07-2021-TJ3DDB7UXRC4TF23FRAB4TBAAY.php

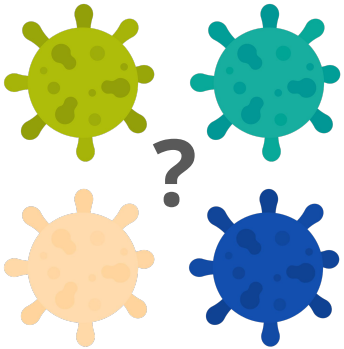
Goal of the “Project”



Sample

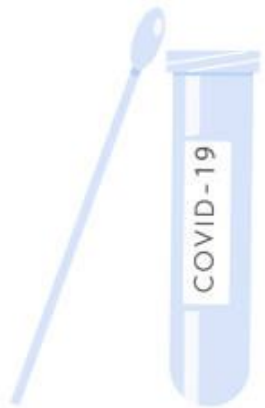


Workflow

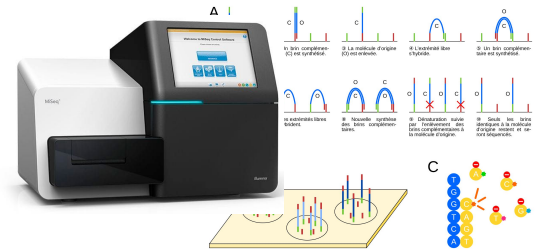


Variant

Biology 101



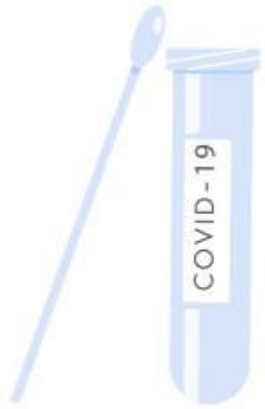
Sample



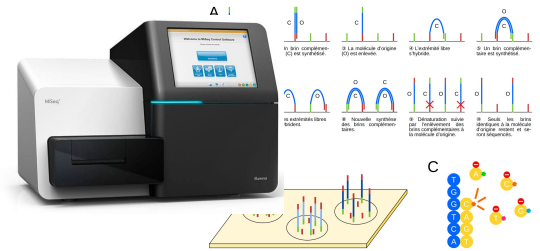
Sequencer

Genetic Code

Biology 101

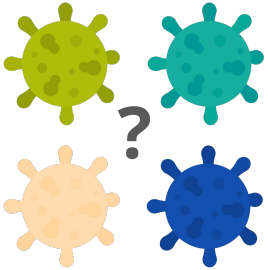


Sample

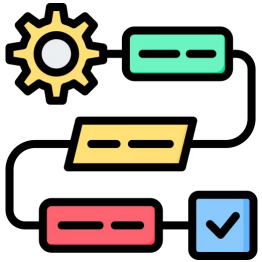


Sequencer

Genetic Code

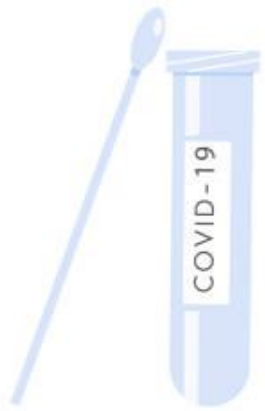


Variant

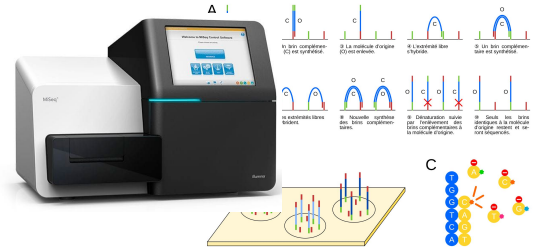


Workflow

Biology 101

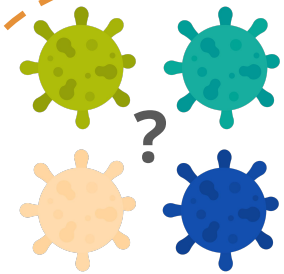


Sample

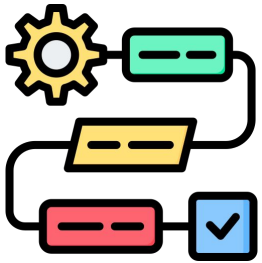


Sequencer

Genetic Code

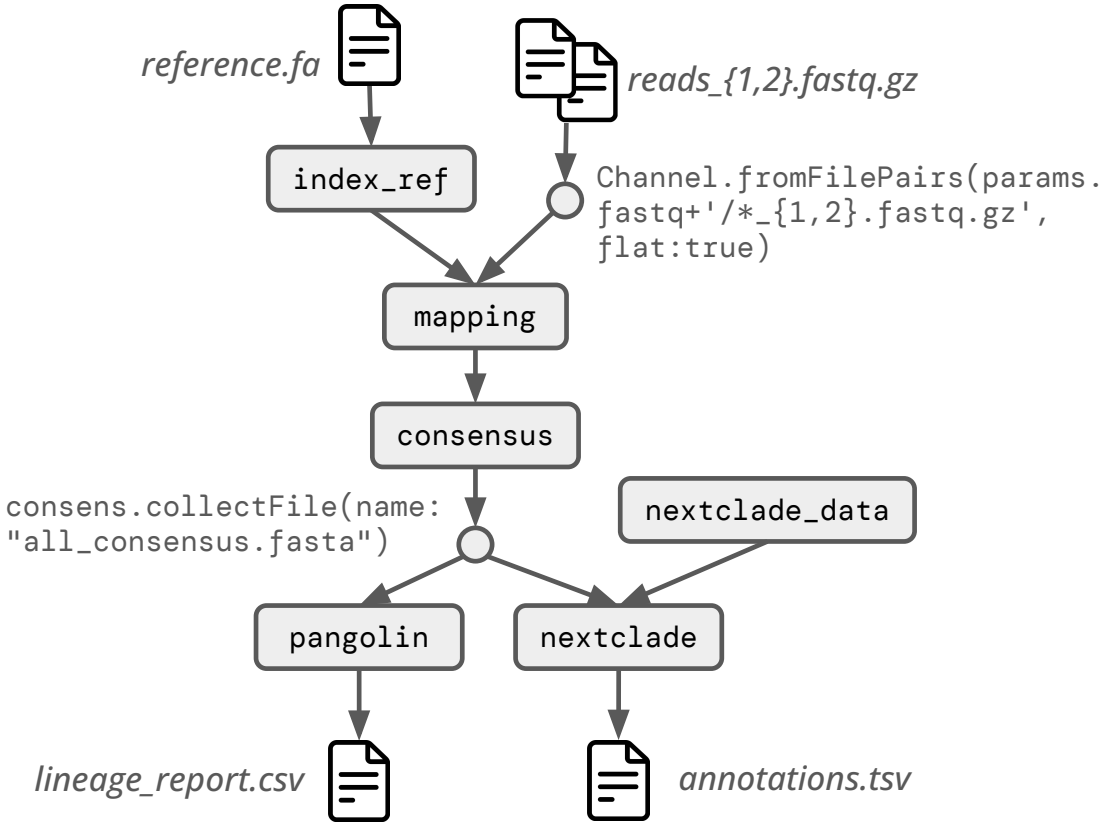


Variant



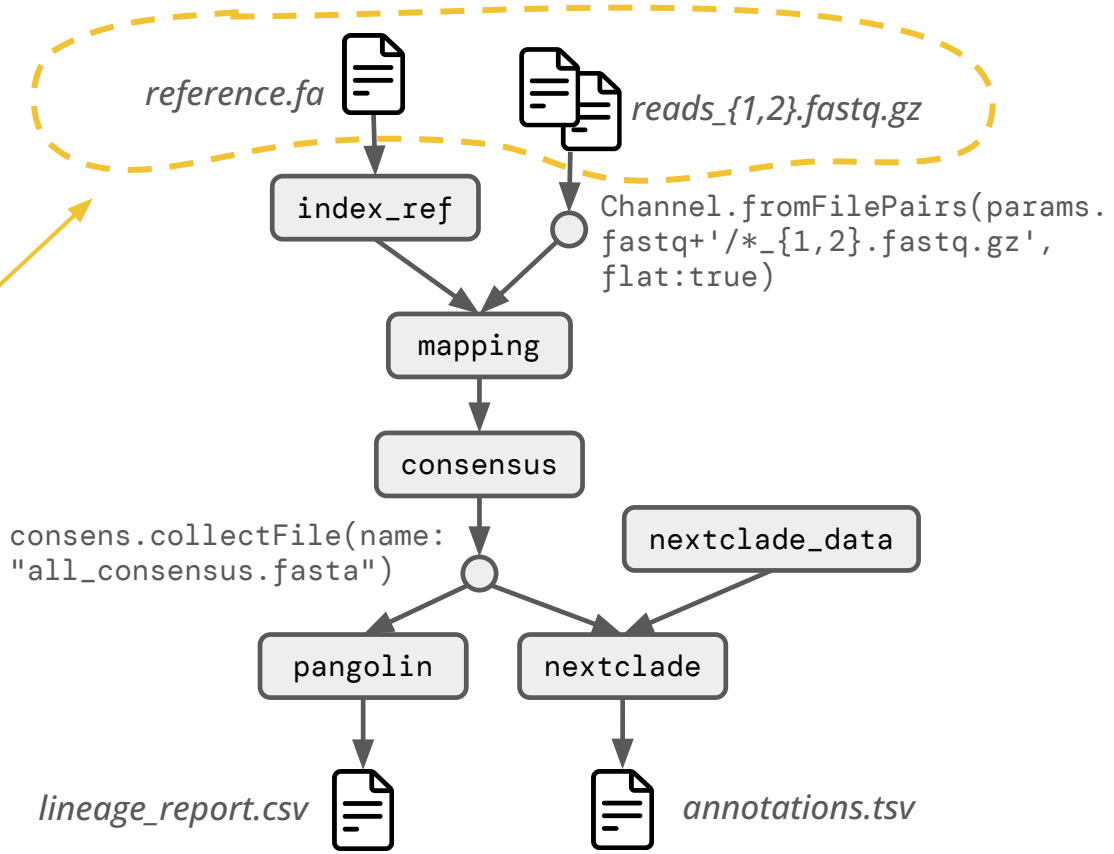
Workflow

The Workflow



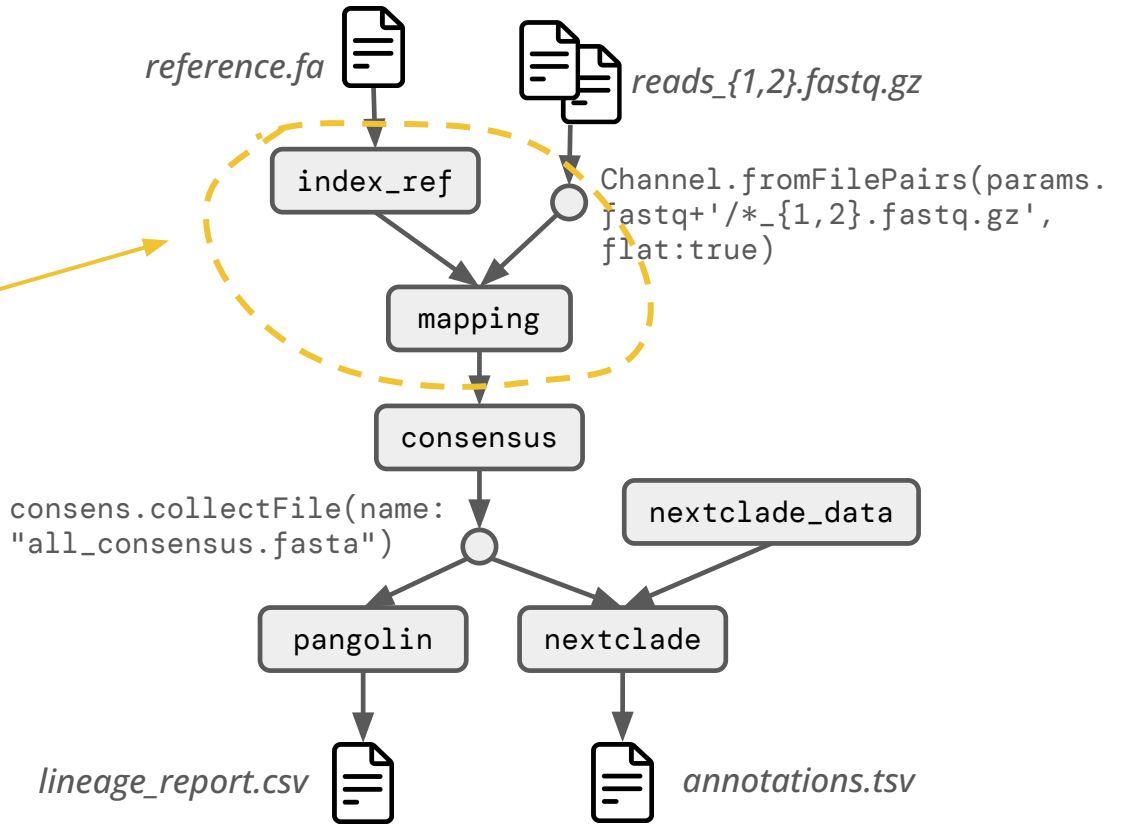
The Workflow

- Inputs:
  - Two compressed fastq files from the SARS-CoV2 sample
  - The reference genome to map



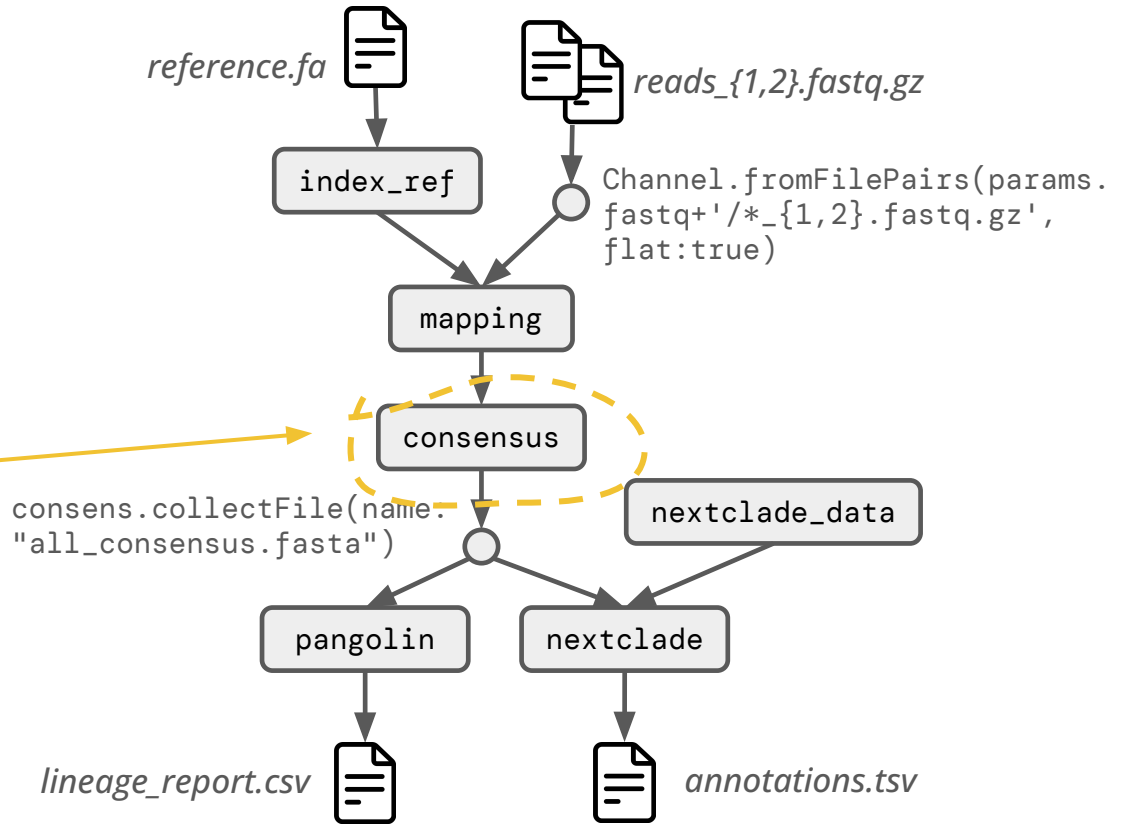
The Workflow

- Step 1:
  - Mapping the reads on a reference genome



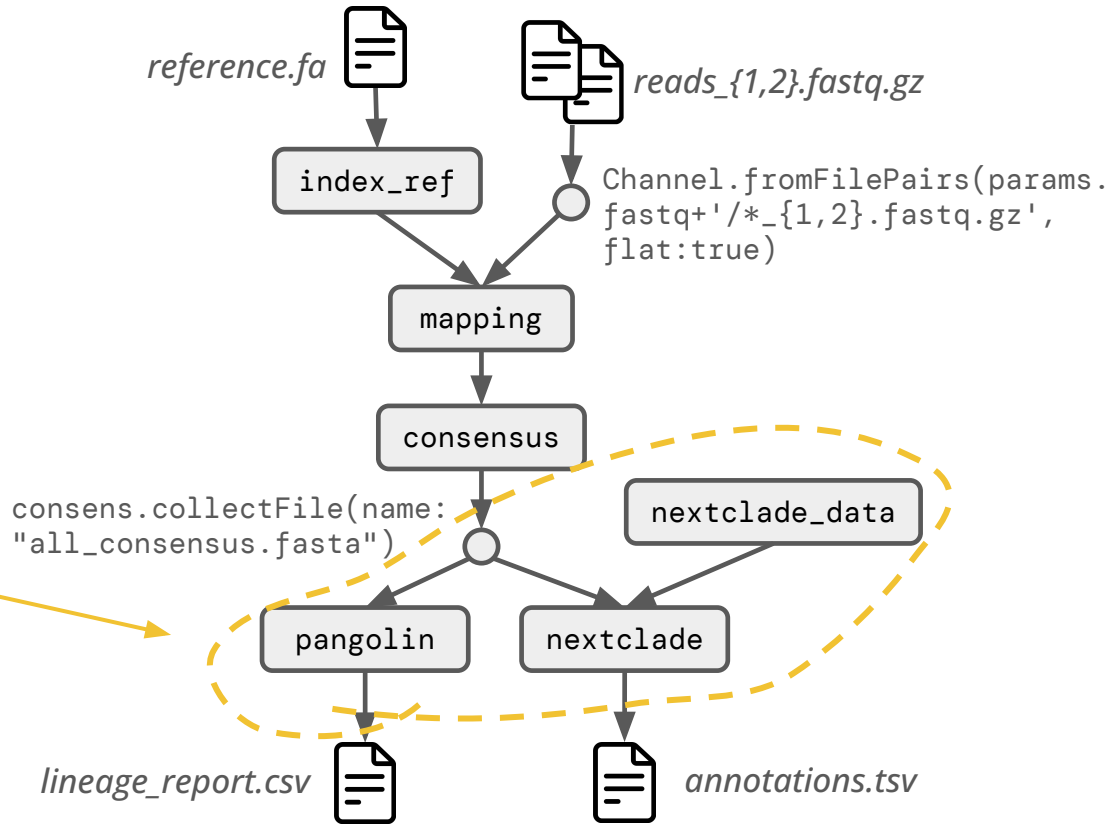
The Workflow

- Step 2:
  - Building consensus sequence



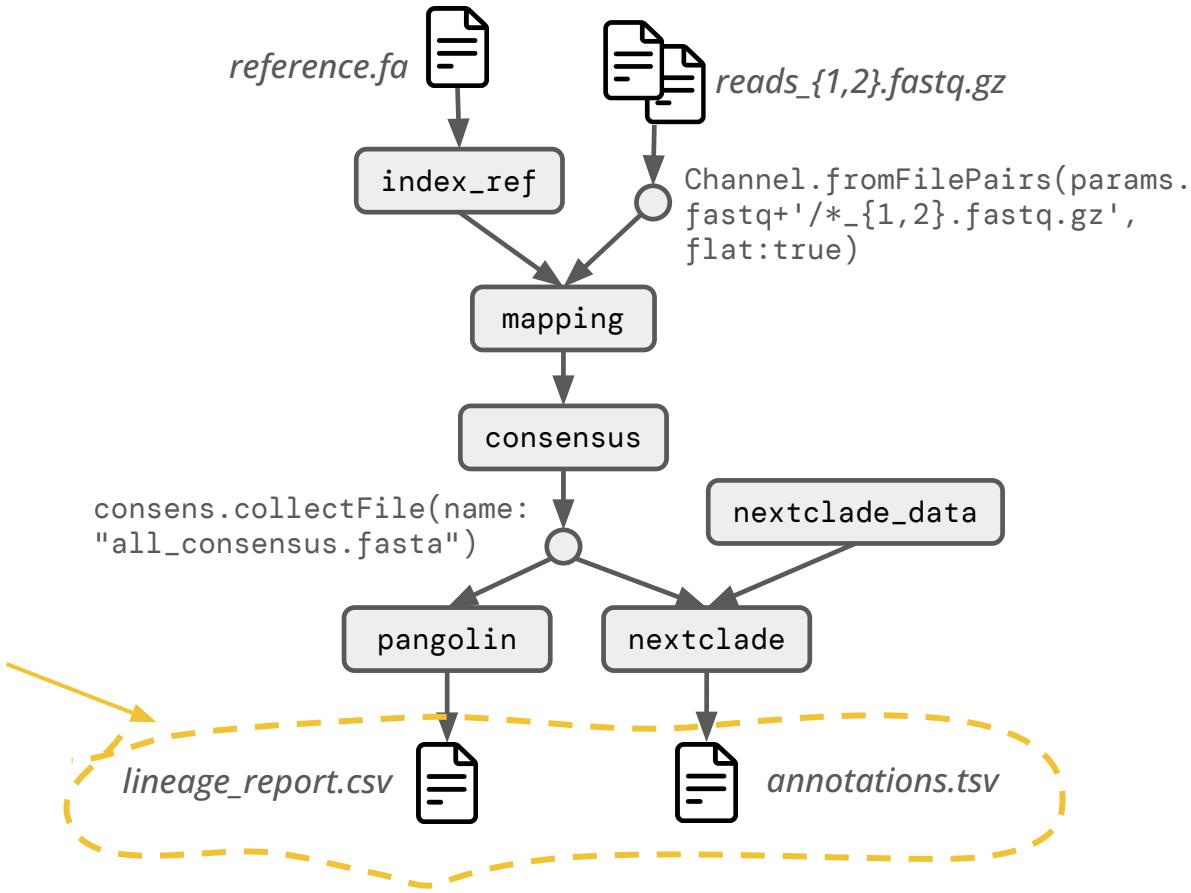
The Workflow

- Step 3:
  - Detecting clade



The Workflow

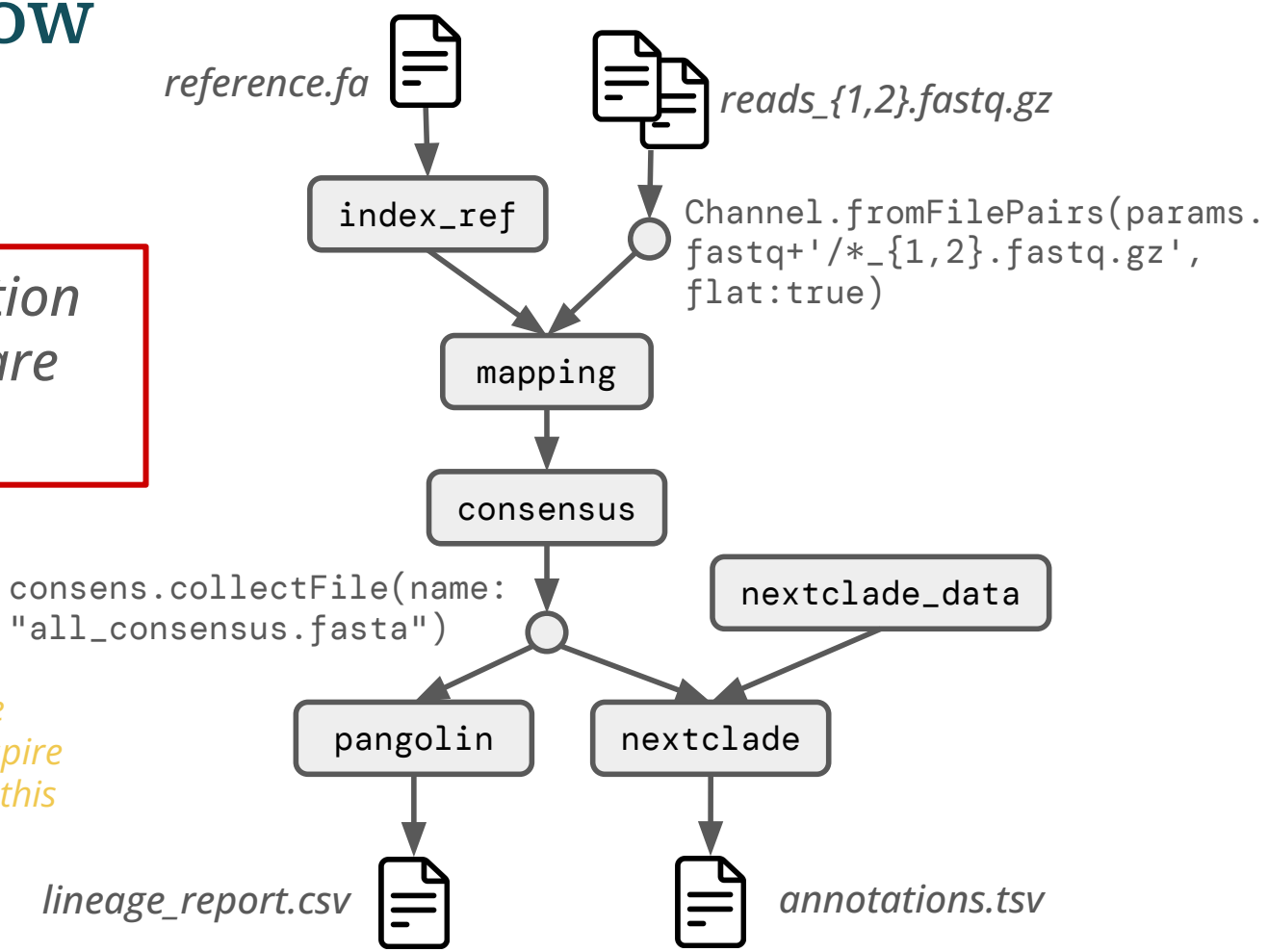
- Outputs:
  - annotations.tsv
  - lineage_report.csv



The Workflow

It's full specification and input data are described here

Use the correction of the use-case workflow to inspire your implementation of this workflow



1. Introduction to (Nextflow) Workflows (~1h)
    a. Motivational Example
    b. Objectives
    c. Scientific Workflows
    d. Software Containers
    e. Nextflow Workflows
    f. Presentation of the “Project”
2. **Practical Work (~1h30)**
3. Reproducibility Consensus (~30min)

1. Introduction to (Nextflow) Workflows (~1h)
    a. Motivational Example
    b. Objectives
    c. Scientific Workflows
    d. Software Containers
    e. Nextflow Workflows
    f. Presentation of the “Project”
2. Practical Work (~1h30)
3. **Reproducibility Consensus (~30min)**

Reproducibility Consensus

- After implementing and running the workflow described above

- Please send the results of the workflow (the **`annotations.tsv`** as well as the **`lineage_report.csv`** file) to george.marchment[at]universite-paris-saclay.fr