Sequana: a set of Snakemake NGS pipelines

Dimitri Desvillechabrol, Rachel Legendre, Mélissa Cardon
and Thomas Cokelaer

September 25th 2017, Institut Curie, Paris

Introduction

Motivation

Development driven by the Biomics Pole at Pasteur Institute, which involves many aspects of NGS including :

https://research.pasteur.fr/en/team/biomics/

- De novo and targeted sequencing of viruses, prokaryotes and eukaryotes
- Variant (SNP, indel, large rearrangements) detection
- Human and Mouse SNP detection by array
- Transcriptional analysis (RNA-Seq) for both prokaryotes and eukaryotes
- 16S and deep-sequencing metagenomic studies (mouse, human, and other environments)
- Bottom-up whole proteomic analysis and quantification
- Analysis of a wide range of post-translational modifications
- Determination of the dynamics of protein complexes.
- Epigenetics (CHIP-Seq, methylation studies)
- Projects involving two or more techniques (i.e. proteogenomics, single-cell DNA/RNA analysis)

How ?

A glue language, a scientific language

A pipeline framework mixing Python and Makefile

Köster, Johannes and Rahmann, Sven. Snakemake - A scalable bioinformatics workflow engine. Bioinformatics 2012.

Dedicated standalone such as genome coverage characterisation.

D. Desvillechabrol, C. Bouchier, S. Kennedy, T. Cokelaer Detection and characterization of low and high genome coverage regions .... BioRxiv https://doi.org/10.1101/092478

Snakemake

What is snakemake ?

- Clusters can be used with minimum efforts (no intrusive code)
- Workflows can be run from or up to a given rule
- Nice logging system to follows the status
- Suspend / Resume
- Various code can be integrated: R, bash, and of course Python

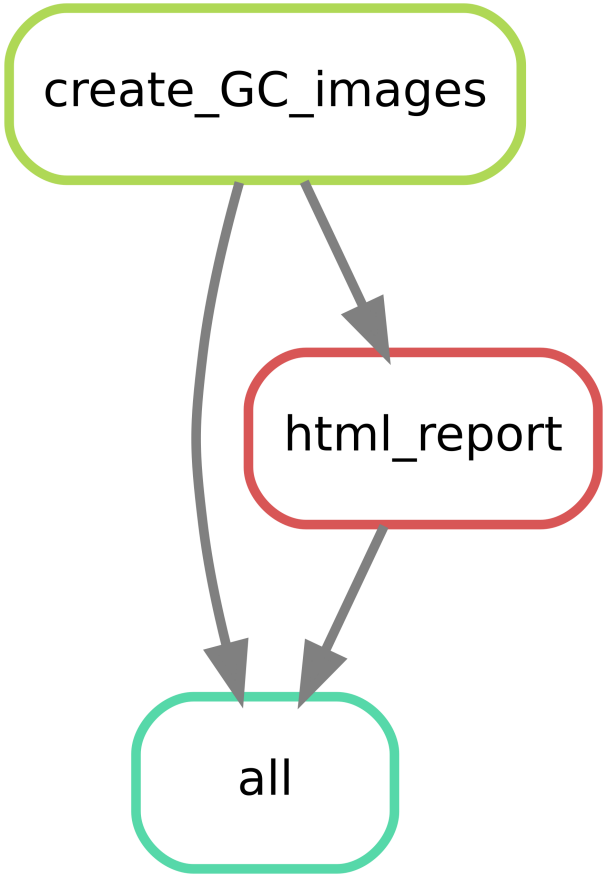
Pipeline example: minimalist (GC content)

```
# list of FastQ files without extension
files = ["A", "B", ....]

rule all:
    input: expand("{data}.png", data=files)
        "index.html"

rule create_GC_images:
    input: "{data}.fastq.gz"
    output: "{data}.png"
    run: # CODE TO COMPUTE IMAGES

rule html_report:
    input: expand("{data}.png", data=files)
    output: "index.html"
    run: # CODE TO CREATE HTML
```



Execution

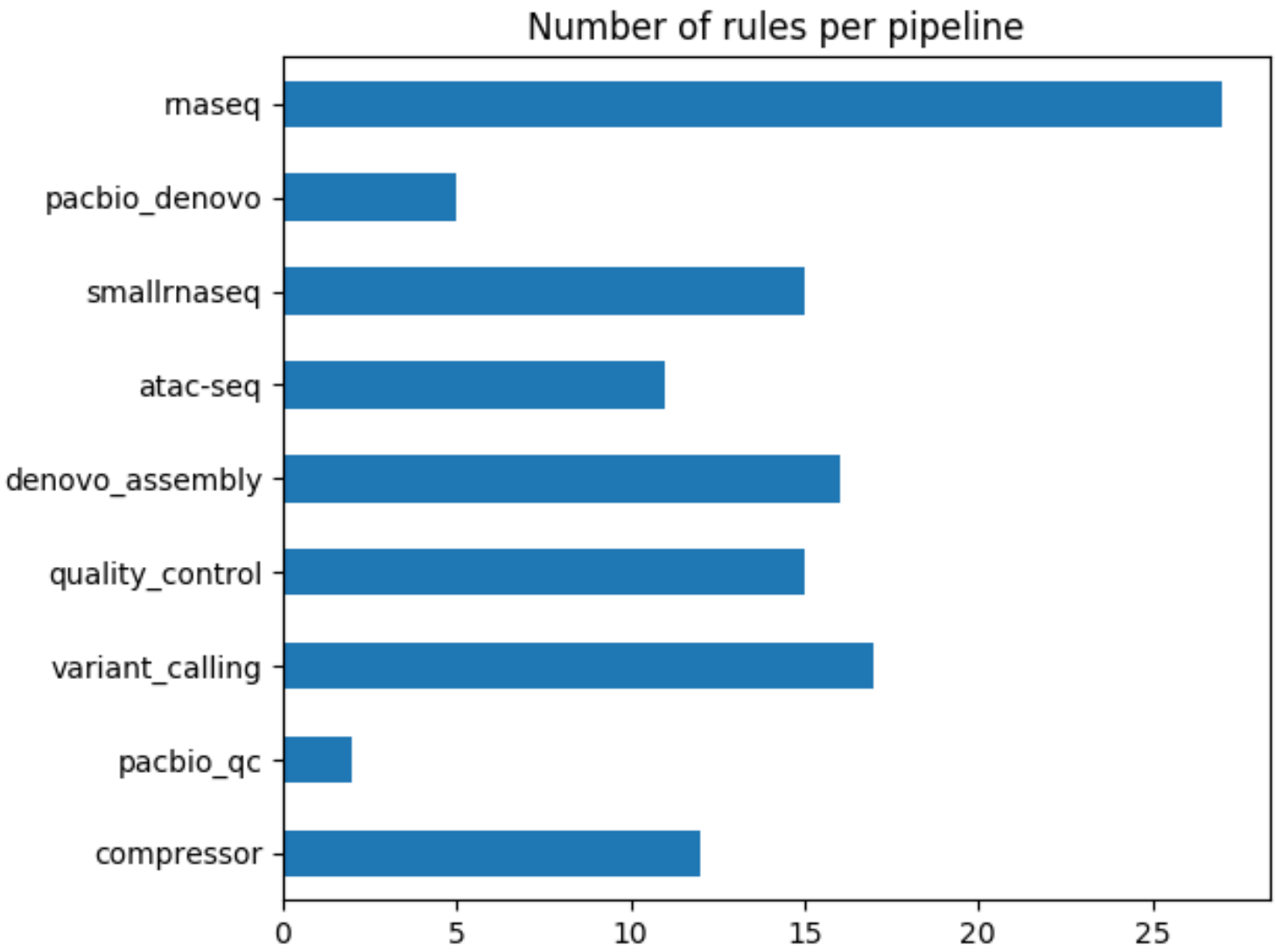
From a shell:

```
snakemake -s gc_minimalist.rules
```

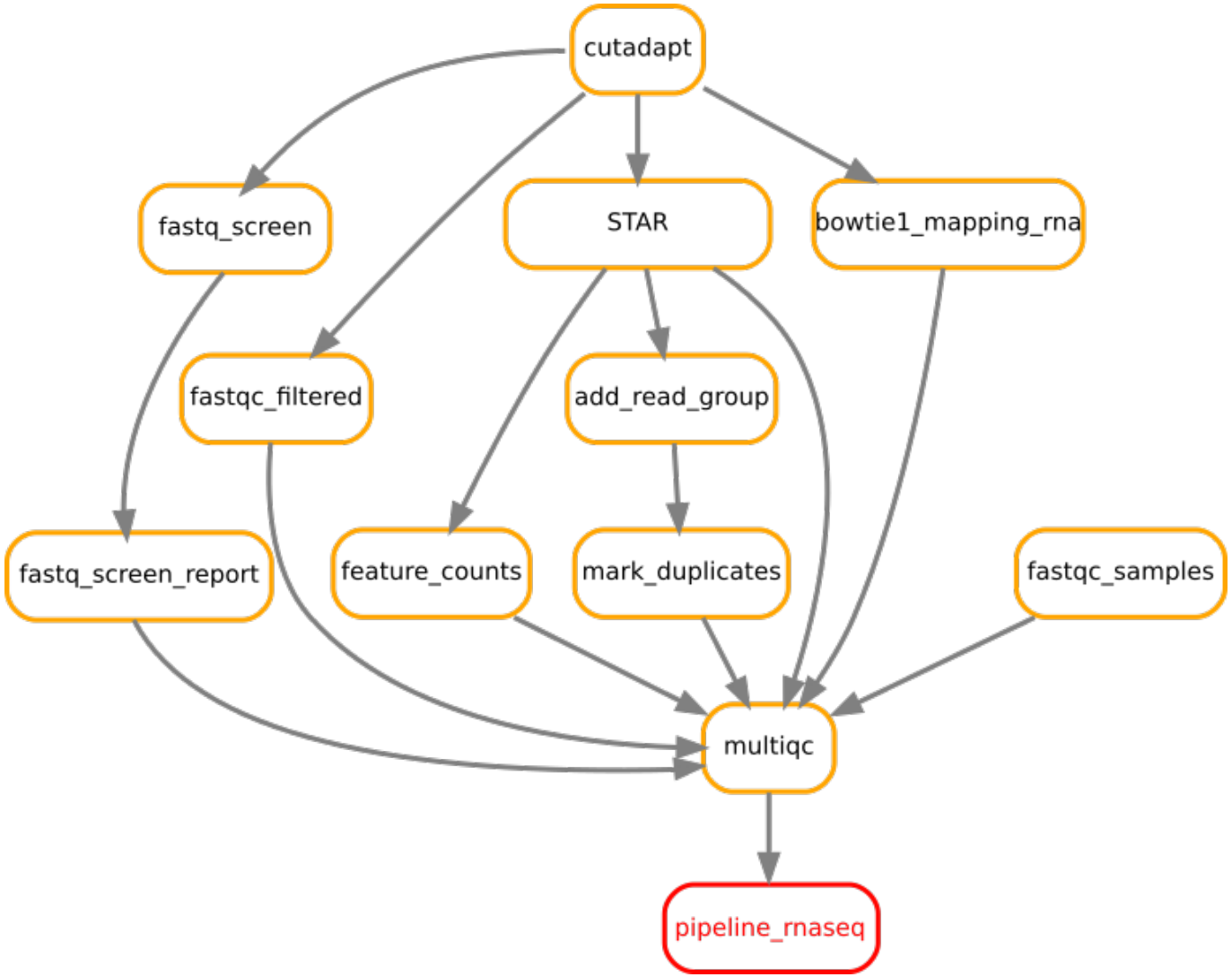
More options if a configuration file is required, or execution is on a cluster, or . . . something goes wrong.

Sequana

Pipelines available in Sequana



Pipeline RNA-seq



Markduplicates rule

```
rule mark_duplicates:
    """DOCSTRING"""
    input:
        __mark_duplicates__input
    output:
        bam = __mark_duplicates__output,
        metrics = __mark_duplicates__metrics
    log:
        out = __mark_duplicates__log_std,
        err = __mark_duplicates__log_err
    params:
        remove = config["mark_duplicates"]["remove"],
        tmpdir = config["mark_duplicates"]["tmpdir"]
    shell:
        """
        (picard MarkDuplicates I={input} O={output.bam} \
            M={output.metrics} REMOVE_DUPLICATES={params.remove} \
            TMP_DIR={params.tmpdir} && samtools index {output.bam}) \
            > {log.out} 2> {log.err}
        """
```

Snakefile

```
# Mark duplicates
if config["mark_duplicates"]["do"]:
    __mark_duplicates__input = __add_read_group__output
    __mark_duplicates__output = manager.getname(
        "mark_duplicates", ".bam")
    __mark_duplicates__metrics = manager.getname(
        "mark_duplicates", ".metrics")
    __mark_duplicates__log_std = manager.getlogdir(
        'mark_duplicates_stdout.logs')
    __mark_duplicates__log_err = manager.getlogdir(
        'mark_duplicates_stderr.logs')
    include: sm.modules["mark_duplicates"]
    expected_output.extend(expand(__mark_duplicates__output,
        sample=manager.samples))
```

YAML configuration file

```
###############################################################################
# mark_duplicates (picard-tools) allows to mark PCR duplicate in
# BAM files
#
# :Parameters:
#
# - do: if unchecked, this rule is ignored. Mandatory for RNA-SeQC
#       tool.
# - remove: If true do not write duplicates to the output file
#           instead of writing them with appropriate flags set.
#           Default value: false. This option can be set to
#           'null' to clear the default value.
#           Possible values: {true, false}
# - tmpdir: write tempory file on this directory
#           (default /tmp/)
#
mark_duplicates:
    do: yes
    remove: no
    tmpdir: "/local/scratch/"
```

Using command line

- One command line to initiate the pipeline

Shell

```
sequana --pipeline rnaseq \
        --input-directory path/to/sample/ \
        --reference sequence.fasta \
        --output-directory analysis/
```

- The sequana executable creates a directory with the project name
- The directory contains all the necessary files (config, snakefile)

Sequanix

Sequanix: a standalone application in PyQt

Sequanix

- Users do not want to see the Snakefile
- Developers do not want users to see the Snakefile

Sequanix: a standalone application in PyQt

Sequanix

- Users do not want to see the Snakefile
- Developers do not want users to see the Snakefile
- Users do not want to edit the configuration file manually
- Developers do not want users to edit the configuration file manually

Sequanix: a standalone application in PyQt

Sequanix

- Users do not want to see the Snakefile
- Developers do not want users to see the Snakefile
- Users do not want to edit the configuration file manually
- Developers do not want users to edit the configuration file manually
- We want a GUI that works on a local computer or on clusters.

Sequanix: a standalone application in PyQt

Sequanix

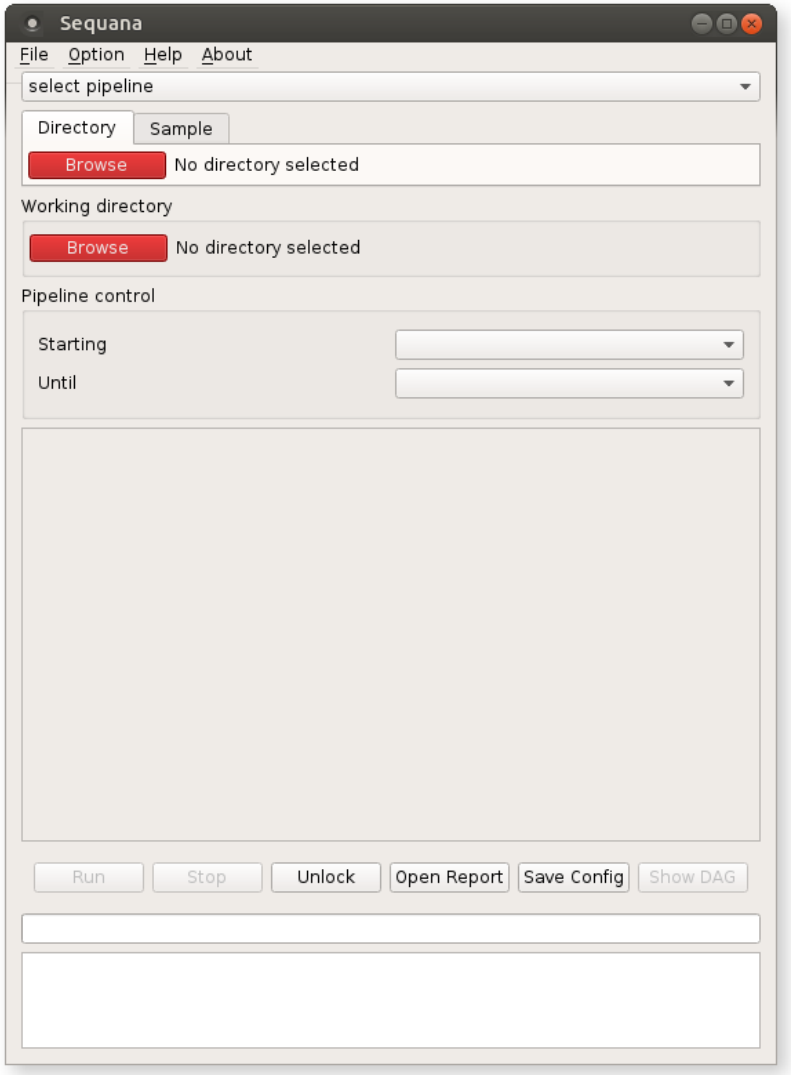
- Users do not want to see the Snakefile
- Developers do not want users to see the Snakefile
- Users do not want to edit the configuration file manually
- Developers do not want users to edit the configuration file manually
- We want a GUI that works on a local computer or on clusters.
- Sequana developers want to expose their pipelines dynamically

Sequanix: a standalone application in PyQt

Sequanix

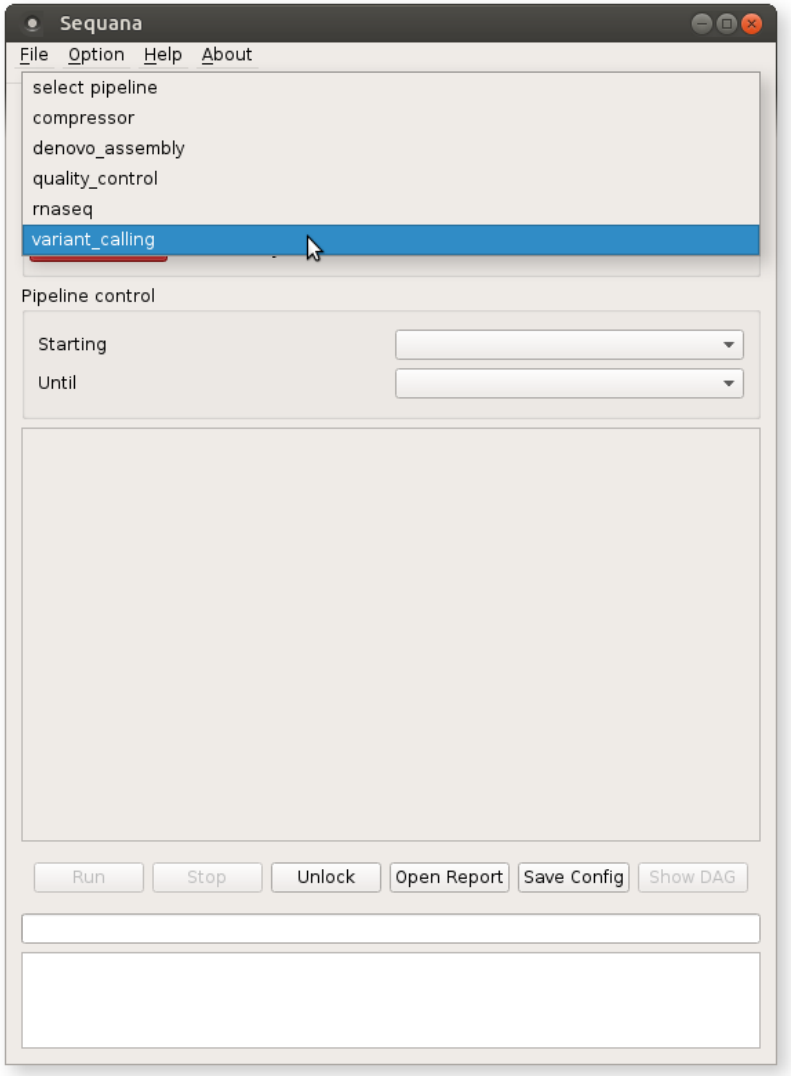
- Users do not want to see the Snakefile
- Developers do not want users to see the Snakefile
- Users do not want to edit the configuration file manually
- Developers do not want users to edit the configuration file manually
- We want a GUI that works on a local computer or on clusters.
- Sequana developers want to expose their pipelines dynamically
- Snakemake developers want to use Sequanix ;-)

GUI to simplify the usage of Snakemake pipeline



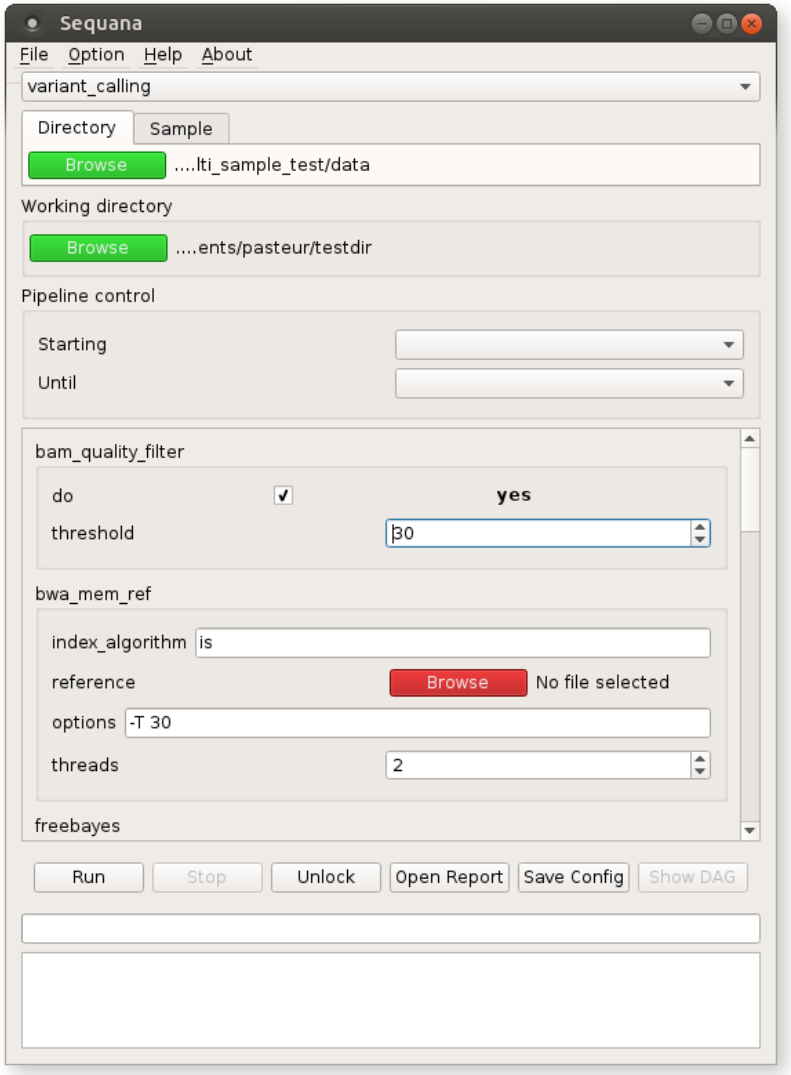
- Interface developed with PyQT5 and Python
- Wrap our snakemake pipelines to ease the usage
- Usable on our cluster, which allows X11

GUI to simplify the usage of Snakemake pipeline



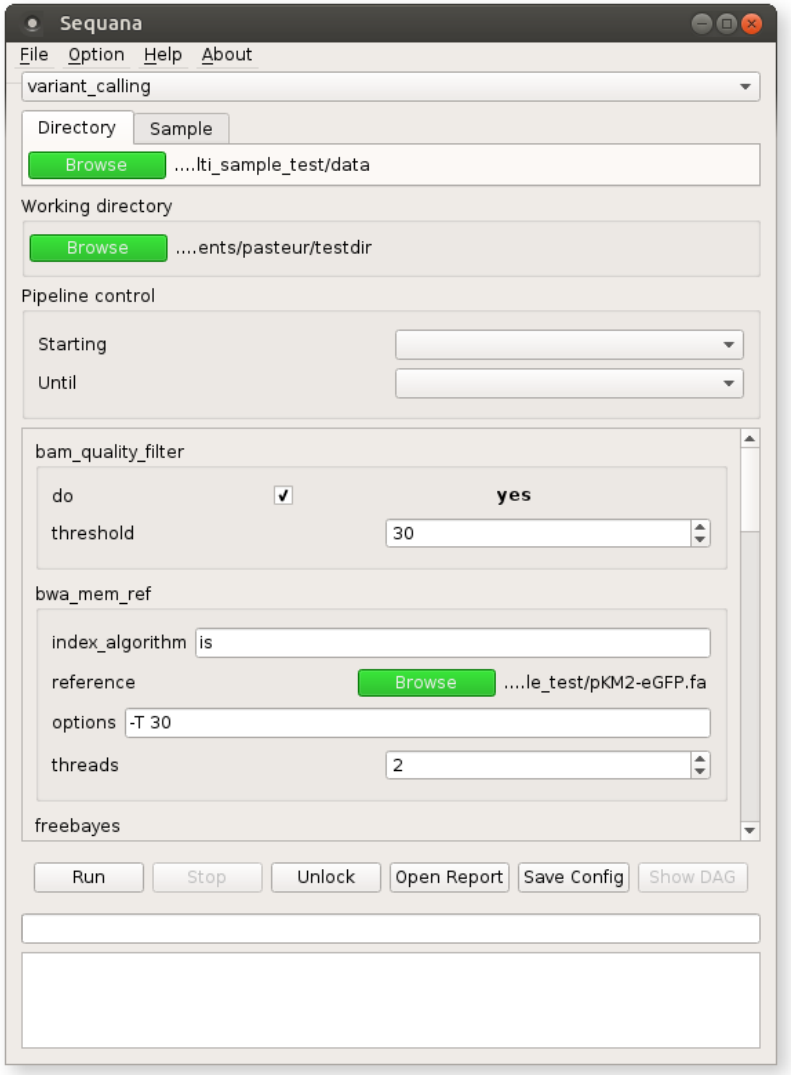
1. Choose a pipeline

GUI to simplify the usage of Snakemake pipeline



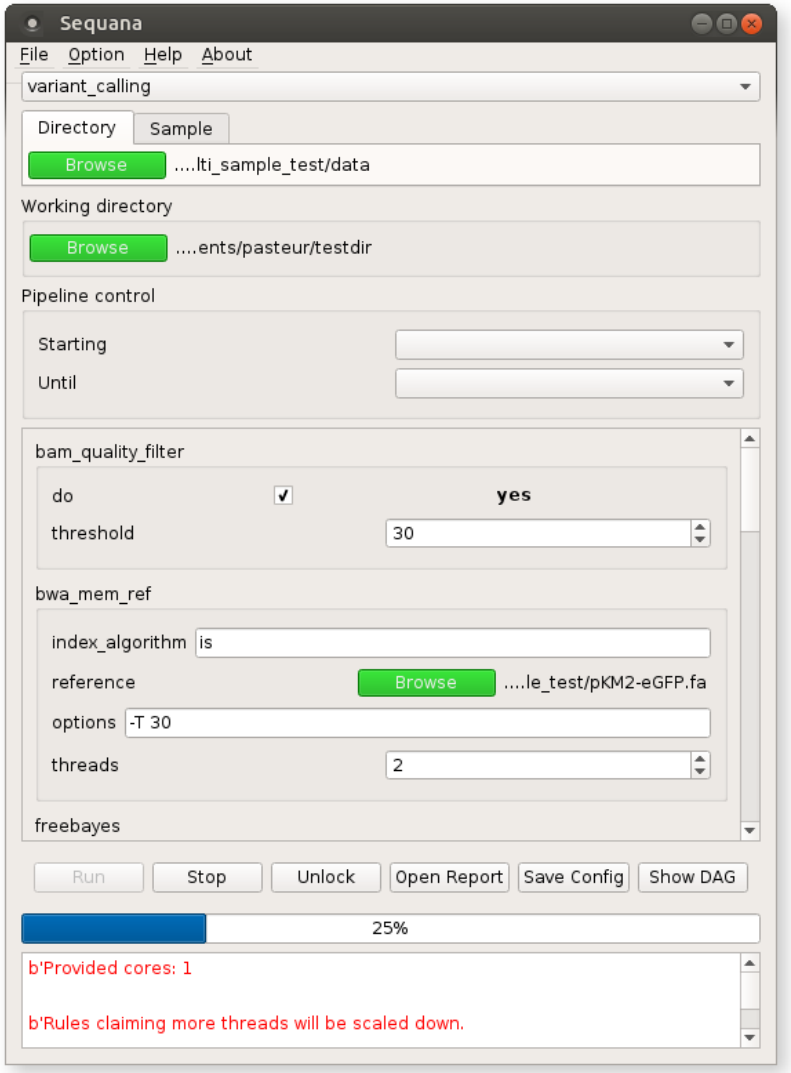
1. Choose a pipeline
2. Set input and output

GUI to simplify the usage of Snakemake pipeline



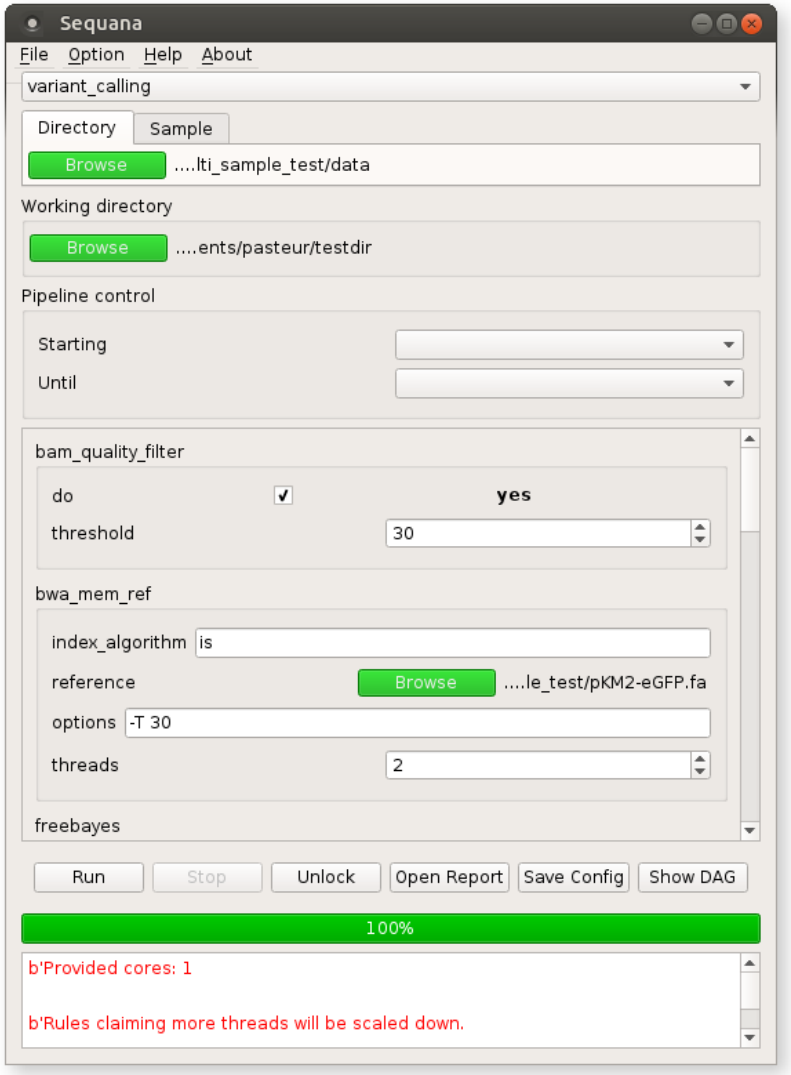
1. Choose a pipeline
2. Set input and output
3. Fill the config form

GUI to simplify the usage of Snakemake pipeline



1. Choose a pipeline
2. Set input and output
3. Fill the config form
4. Run the pipeline

GUI to simplify the usage of Snakemake pipeline



1. Choose a pipeline
2. Set input and output
3. Fill the config form
4. Run the pipeline
5. Finished !

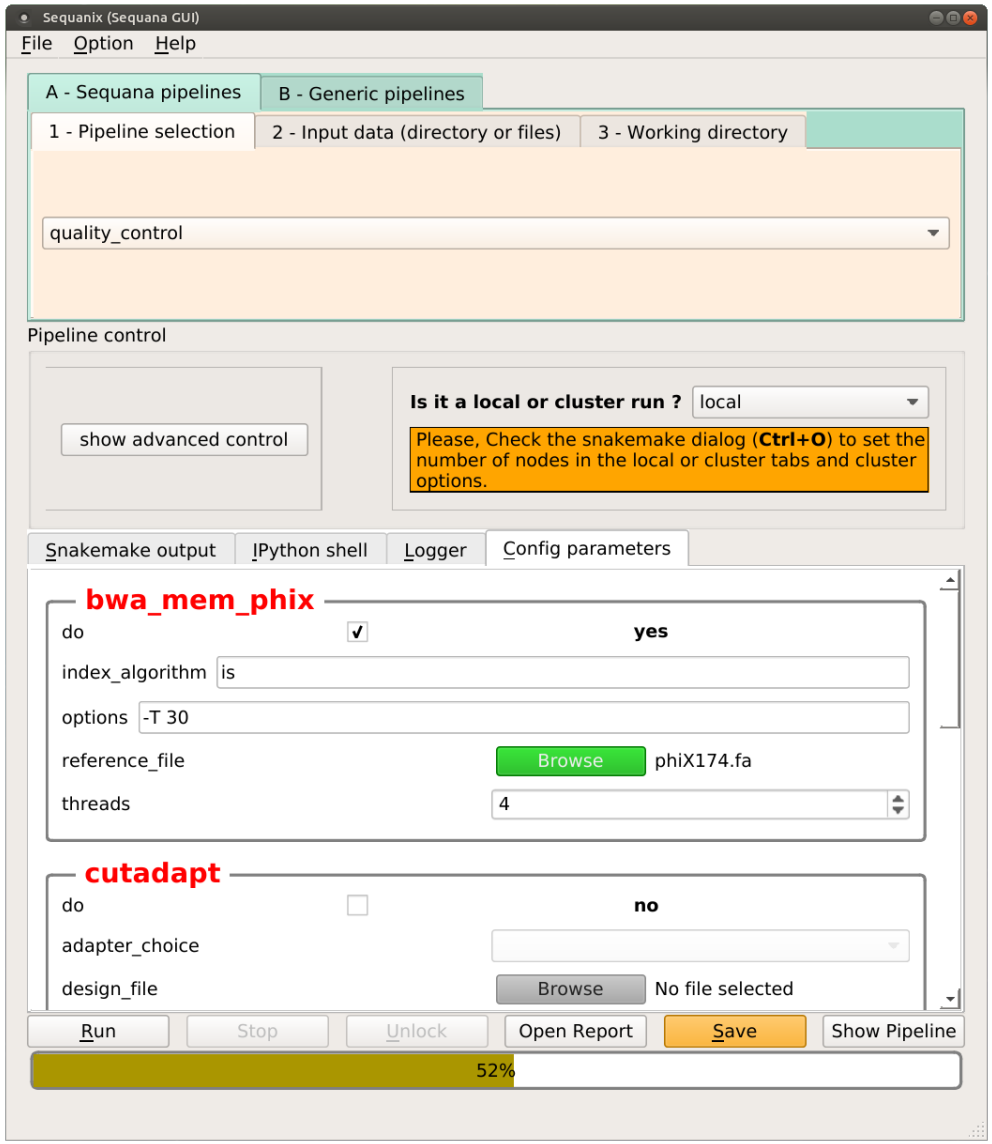
Automatic tooltips

BWA used to remove a contaminant

Parameters:

- do: if unchecked, this rule is ignored
- reference_file: the name of the reference file to be found in the analysis directory. If set to phiX174.fa, it is downloaded automatically from Sequana, otherwise you will need to copy it yourself in the working directory.
- index_algorithm: the BWA index algorithm
- options: any options recognised by BWA tool
- threads: number of threads to be used

Last version of Sequanix



Desvillechabrol, D., Legendre, R., Rioualen, C., Bouchier, C., van Helden, J., Kennedy, S., & Cokelaer, T. (2017). Sequanix: A Dynamic Graphical Interface for Snakemake Workflows. bioRxiv, 162701.

Continuous Integration

Versioning, Test and Documentation

https://github.com/sequana/sequana

Continuous Integration on Travis with 100 tests with 75% coverage

Uses Sphinx (RST syntax) to document the source code and provides user guide.

Updated after each commits on sequana.readthedocs.io

Summary

Summary

- Sequana is a useful framework built on:
  - Snakemake provides a very convenient framework to build pipelines.
  - Python language provide an API to ease the pipeline design.
  - Continous integration able to create high quality tool.
  - Sequanix provides a GUI to run Snakemake pipelines for all type of users.
- If you want to test it:
  - Documentation: http://sequana.readthedocs.io/
  - Singularity: shub://sequana/sequana:master