

Pipeliner Overview

This document contains two sections:

- A) Introduction to Pipeliner
- B) General instruction on launching Pipeliner

A) Introduction to Pipeliner

Pipeliner provides access to some of the NGS data analysis pipelines used by CCBR on the NIH *Biowulf* Linux Cluster. *Pipeliner* allows the user to select a set of data files, such as fastq sequence reads, and process them via a sequence of programs that constitute an analysis 'pipeline' to reach an endpoint, such as a QC report, a list of mutations with accompanying annotations or a list of differentially expressed genes/isoforms. The program provides a graphical interface in which pipeline options are configured using pulldowns and text entry fields. Once configured, the pipeline is executed on the *Biowulf* cluster.

B) Launching Pipeliner

B1) Pre-requisites

- *An account on the NIH Biowulf Linux Cluster*

Pipeliner runs on the NIH *Biowulf* cluster and uses a graphical interface. To use the program one must log into *Biowulf* using ssh with X11 packet forwarding. For instance:

➤ `ssh -Y username@biowulf.nih.gov`

- *A data directory containing fastq reads*

For all pipelines, the expected naming convention is as follows:

SAMPLENAME.R1.fastq.gz, SAMPLENAME.R2.fastq.gz

However, if your samples are differently named or you would like to convert original

sample names to something more appropriate for biological comparisons or analysis (e.g., add tissue sources or other identifier), simply place in the same directory as your read files a plain text file named *labels.txt* with the current sample name in the left column and the new sample name in the right column, and read file names will be converted to the appropriate name and format when symlinks are created in your working directory (symlink generation and working directory details are explained in detail below).

- *A new working directory*

This is the directory where all resulting files will be written.

B2) Running *Pipeliner*

B2.1) Once logged in, launch the *Pipeliner* program:

```
>module load ccbrpipeliner  
>ccbrpipe.sh
```

After a delay of a few seconds, the GUI opens:

CCBR Pipeliner

File View Help

Project Information

Project Id (Examples: CCBR-nnn, Labname or short project name)

Email address (Mandatory field: must use @nih.gov email address)

Flow Cell ID (Examples: FlowCellID, Labname, date or short project name)

Global Settings

Genome: Pipeline Family:

Project Description

Enter CCBP Project Description and Notes here.

B2.2) Within the *Pipeliner* GUI, the workflow is as follows:

- Fill in project information box
 - Provide a project ID (this can be any ID you choose, or left to the default),
 - Your full NIH email (i.e., username@biowulf2.nih.gov). Note: This field is mandatory to fill in, and MUST contain an email ending in *.nih.gov.
 - Flowcell ID (or just leave the default)

The screenshot shows the 'CCBR Pipeliner' application window. It has a menu bar with 'File', 'View', and 'Help'. The main section is titled 'Project Information' and contains three input fields with their respective labels and examples:

Field Label	Input Value	Examples / Notes
Project Id	project	(Examples: CCBR- <i>nnn</i> , Labname or short project name)
Email address		(Mandatory field: must use @nih.gov email address)
Flow Cell ID	stats	(Examples: FlowCellID, Labname, date or short project name)

B2.3) Fill/Select global options

On this box, you can select the genome you wish to use under annotation set (i.e., select between mouse mm10 and human hg19) and specify the pipeline you want to run (exome-seq, runseq ...). You can also write a description of the project

The screenshot shows the 'Global Settings' and 'Project Description' sections of the CCBP Pipeliner application. The 'Global Settings' section has two dropdown menus: 'Genome' (set to 'hg19') and 'Pipeline Family' (set to 'Select a pipeline'). The 'Project Description' section has a text area for entering the project description and a list of pipeline options.

Global Settings

Genome	Pipeline Family
hg19	Select a pipeline

Project Description

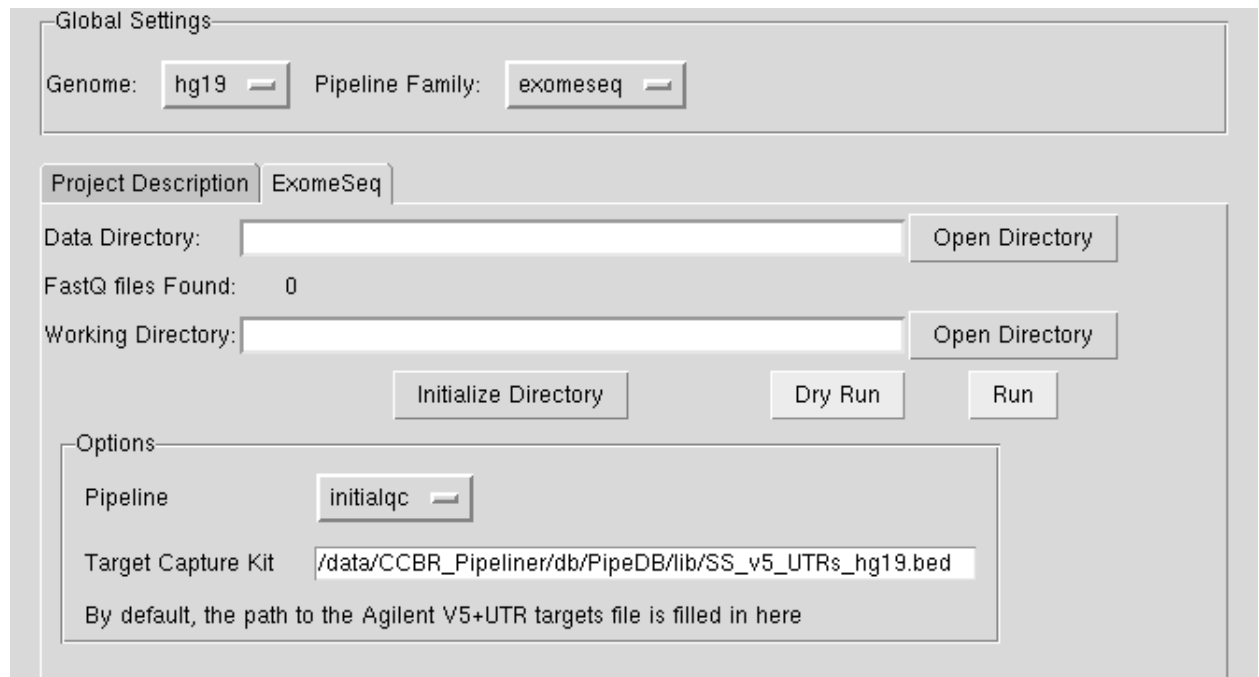
Enter CCBP Project Description and

- exome-seq
- runseq
- genome-seq
- mirseq
- chipseq

B2.4) Fill/Select pipeline-specific options

See pipeline-specific tutorials for step-by-step instructions detailing this step.

Once you select a pipeline, a new pipeline tab will be added right to the project description tab. For instance, if you choose the “exomeseq” pipeline you have the following Interface



The screenshot displays the 'Global Settings' window for the 'ExomeSeq' pipeline. At the top, 'Genome' is set to 'hg19' and 'Pipeline Family' is set to 'exomeseq'. Below this, the 'Project Description' tab is active, showing 'ExomeSeq'. The 'Data Directory' field is empty, with 'FastQ files Found' at 0 and an 'Open Directory' button. The 'Working Directory' field is also empty, with an 'Open Directory' button. Below these are three buttons: 'Initialize Directory', 'Dry Run', and 'Run'. An 'Options' section at the bottom contains a 'Pipeline' dropdown set to 'initialqc', a 'Target Capture Kit' text field with the path '/data/CCBR_Pipelinier/db/PipeDB/lib/SS_v5_UTRs_hg19.bed', and a note: 'By default, the path to the Agilent V5+UTR targets file is filled in here'.

Each pipeline has its specific parameters however for all of them you need to specify the location of raw data (data directory) as well as the location of the pipeline output (working directory). Please set the data directory (the directory containing all of your read files, either already in the name format `SampleName.R1.fastq.gz` `SampleName.R2.fastq.gz`, or with an accompanying *labels.txt* file to indicate read file names should be renamed/reformatted when symlinks are generated in the working directory). Also, give the full path to a working directory that does not exist, but will be created by Pipelinier, in which all Pipelinier files and result files will be generated:

CCBR Pipeliner

File View Help

Project Information

Project Id: Project_01 (Examples: CCBR-nnn, Labname or short project name)

Email address: user@mail.nih.gov (Mandatory field: must use @nih.gov email address)

Flow Cell ID: stats (Examples: FlowCellID, Labname, date or short project name)

Global Settings

Genome: hg19 Pipeline Family: exomeSeq

Project Description ExomeSeq

Data Directory: /data/CCBR/example/data Open Directory

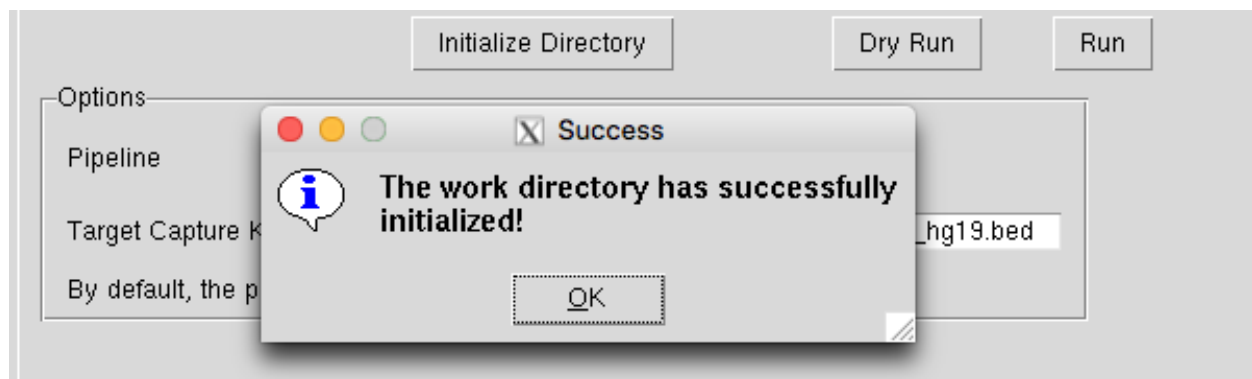
FastQ files Found: 0

Working Directory: /data/CCBR/example/pipeline_out Open Directory

The pipeline displays automatically the number of fastq files found in the working directory

B2.5) Initialize, test and run the pipeline

B2.5.a) Initialize the working directory—Clicking this button makes a few subdirectories within it, and populates these with a number of files needed to run the pipelines.



B2.5.b) Perform a dry run to verify that the pipeline is configured properly—a successful dry run will display all tasks to be run and it will look something like this:

[illegible]

B2.5.c) Launch the pipeline--this submits the pipeline job to the *Biowulf* cluster

AND YOU ARE OFF AND RUNNING!

The pipeline will generate an email when it begins running, and will also generate an email when it stops. To ensure it has run to completion, simply navigate to the Reports directory within your working directory, open the `snakemake.log` file in any text editor, and examine the last few lines. This file tracks progress of all jobs in your pipeline, and if the pipeline runs to completion it will look like this:

```

[Mon Jun 6 11:31:41 2016] 6 of 10 steps (60%) done
[Mon Jun 6 11:31:41 2016] rule snpeff:
  input: all.snp.vcf, all.indel.vcf
  output: all.snp.snpeff.vcf, all.indel.snpeff.vcf, stats_summary.snp.html, stats_summary.i
[Mon Jun 6 12:10:21 2016] 7 of 10 steps (70%) done
[Mon Jun 6 12:10:21 2016] rule snpeff_dbnsfp:
  input: all.snp.filter.vcf, all.indel.filter.vcf
  output: all.snp.dbnsfp.vcf, all.indel.dbnsfp.vcf
[Mon Jun 6 12:56:36 2016] 8 of 10 steps (80%) done
[Tue Jun 7 04:28:47 2016] 9 of 10 steps (90%) done
[Tue Jun 7 04:28:47 2016] localrule all_exomeseq_germline:
  input: combined.gvcf, all.snp.dbnsfp.vcf, all.indel.dbnsfp.vcf, full_annot.txt.zip
[Tue Jun 7 04:28:47 2016] 10 of 10 steps (100%) done

```

However, if a job fails for any reason, the log will indicate it:

```

[Thu Sep 22 00:31:05 2016] Will exit after finishing currently running jobs.
[Thu Sep 22 00:32:10 2016] 71 of 182 steps (39%) done
[Thu Sep 22 00:32:10 2016] Will exit after finishing currently running jobs.
[Thu Sep 22 00:36:11 2016] 72 of 182 steps (40%) done
[Thu Sep 22 00:36:11 2016] Will exit after finishing currently running jobs.
[Thu Sep 22 00:40:41 2016] 73 of 182 steps (40%) done
[Thu Sep 22 00:40:41 2016] Will exit after finishing currently running jobs.
[Thu Sep 22 00:46:14 2016] 74 of 182 steps (41%) done
[Thu Sep 22 00:46:14 2016] Will exit after finishing currently running jobs.
[Thu Sep 22 00:53:54 2016] 75 of 182 steps (41%) done
[Thu Sep 22 00:53:54 2016] Will exit after finishing currently running jobs.
[Thu Sep 22 00:56:06 2016] 76 of 182 steps (42%) done
[Thu Sep 22 00:56:06 2016] Will exit after finishing currently running jobs.
[Thu Sep 22 01:04:07 2016] 77 of 182 steps (42%) done
[Thu Sep 22 01:04:07 2016] Will exit after finishing currently running jobs.
[Thu Sep 22 01:05:18 2016] 78 of 182 steps (43%) done
[Thu Sep 22 01:05:18 2016] Will exit after finishing currently running jobs.
[Thu Sep 22 01:07:55 2016] 79 of 182 steps (43%) done
[Thu Sep 22 01:07:55 2016] Will exit after finishing currently running jobs.
[Thu Sep 22 01:50:52 2016] 80 of 182 steps (44%) done
[Thu Sep 22 01:50:52 2016] Will exit after finishing currently running jobs.
[Thu Sep 22 02:02:07 2016] 81 of 182 steps (45%) done
[Thu Sep 22 02:02:07 2016] Will exit after finishing currently running jobs.
[Thu Sep 22 02:09:21 2016] 82 of 182 steps (45%) done
[Thu Sep 22 02:09:21 2016] Will exit after finishing currently running jobs.
[Thu Sep 22 02:09:22 2016] Exiting because a job execution failed. Look above for error message

```

B3) Configuration and Details

Pipelinr uses a program called Snakemake to manage pipeline workflows.

<https://bitbucket.org/snakemake/snakemake/wiki/Home>

Snakemake, in turn, accepts a configuration file formatted in JSON (Javascript Object Notation).

B4) The principal configuration files used by *Pipeliner*

- hg19.json: references for human genome version hg19/GRCh37
- standard-bin.json: paths to programs used in the pipeline
- rules.json: lists of Snakemake rules assigned to named pipelines
- cluster.json: SLURM parameters for each rule (memory requested, time, # cpus)

B5) Backend python programs

- pipeliner2.py: main program, including GUI components
- makeasake.py: called by *Pipeliner* to build Snakefiles required by Snakemake
- stats2html.py: builds reports at end of a pipeline run

B6) Subdirectories generated within working directory after initialization

- Reports: contains scripts for aggregate report generation and aggregate reports created by *Pipeliner*
- QC: contains QC reports generated during various pipeline steps
- Scripts: contains custom scripts used by some pipelines

B7) Pipelines Implemented

- ExomeSeq (both somatic and germline)
- GenomeSeq (germline whole genome variant detection)
- RNASeq
- ChipSeq

B8) Pipelines Planned for Inclusion

- ScRNASeq (single cell RNASeq)
- mirSeq

B9) Organisms supported

- Human (hg19)
- Mouse (mm10)