Pipeliner

This document contains four sections:

A) Introduction to Pipeliner
B) General instruction on launching Pipeliner
C) Specific instructions for executing ExomeSeq/GenomeSeq pipeline
D) Specific instructions for executing RNASeq pipeline

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).

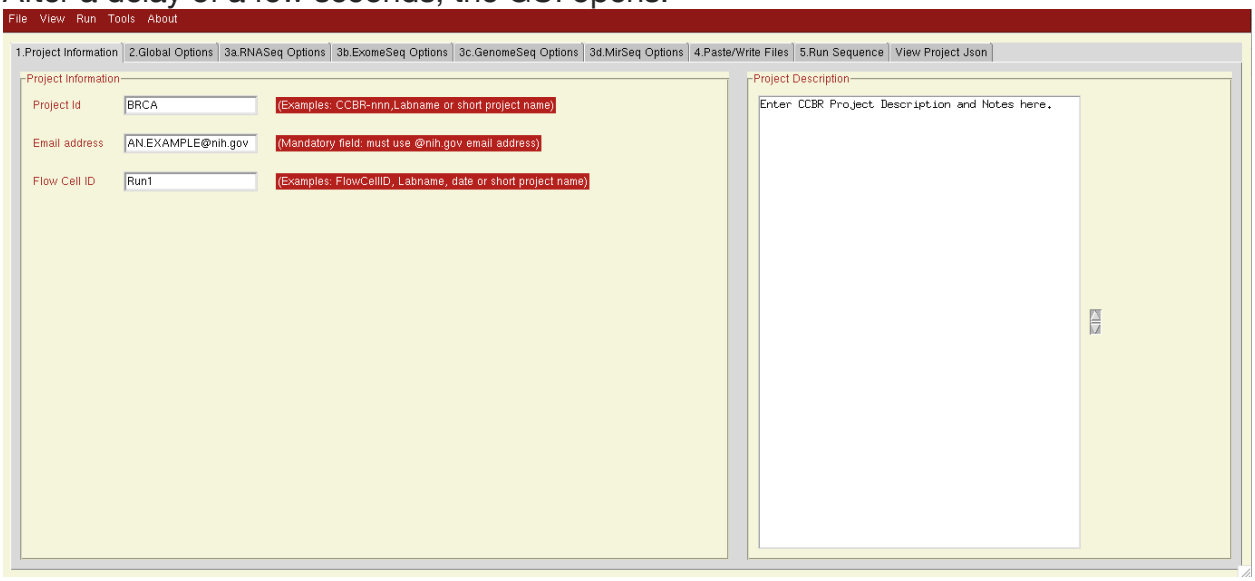
- ***A new working directory***
  This is the directory where all resulting files will be written.

B2) Running *Pipeliner*

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

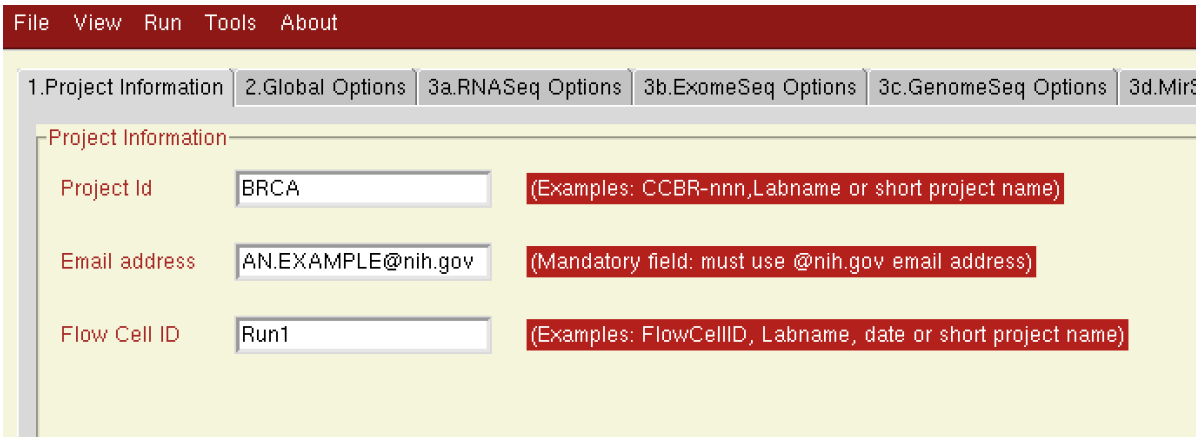
```
>module load ccbrpipeliner
>ccbrpipe.sh
```

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

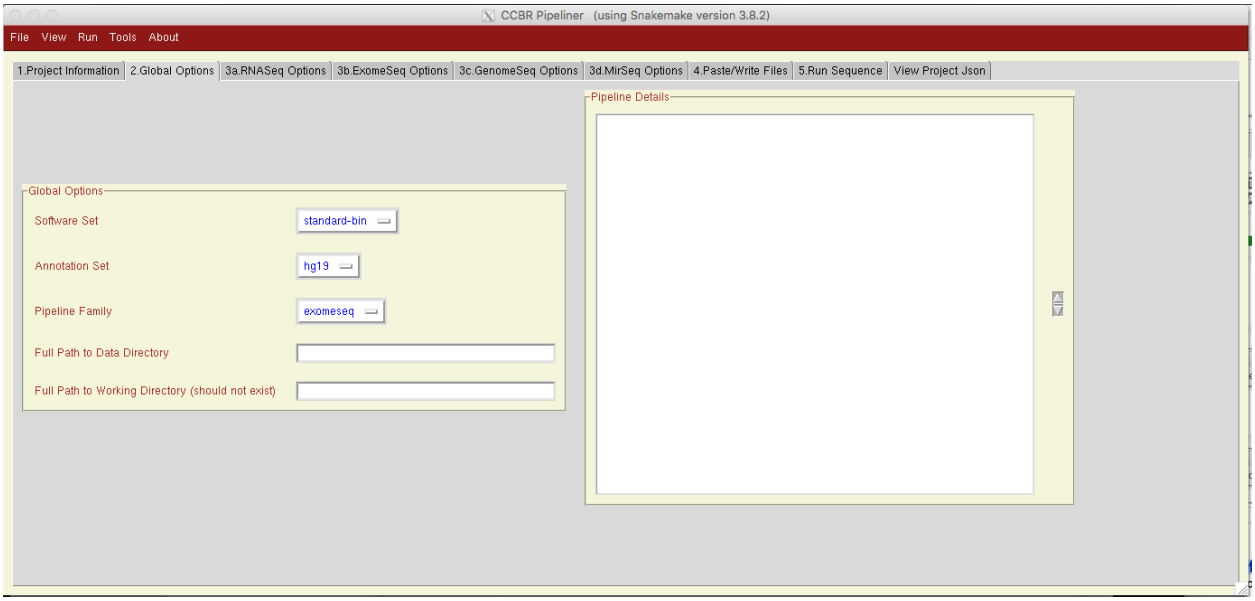


B2.2) Within the *Pipeliner* GUI, the workflow is as follows:

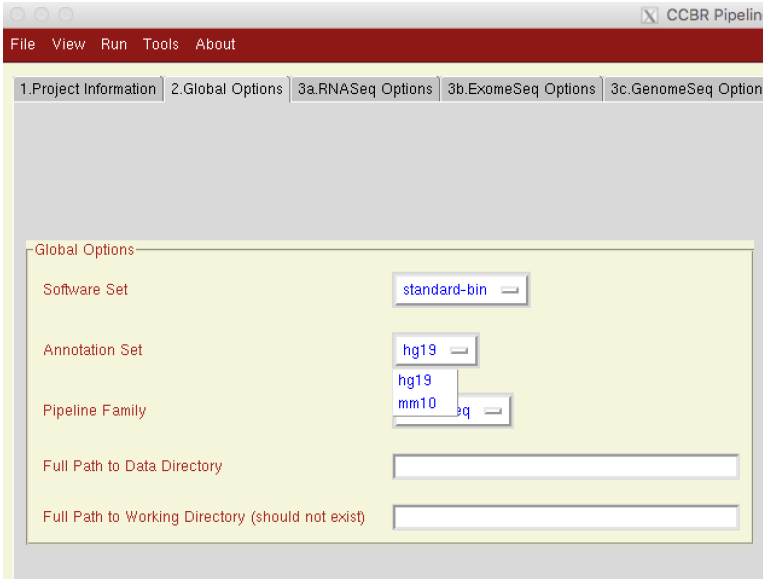
- Fill in project information (tab 1)—
  - Provide a project ID (this can be any ID you choose, or left to the default),
  - Your full NIH email (i.e., username@biowulf2.nih.gov). Note: This field is mandatory to fill in, and MUST contain an email ending in *nih.gov.
  - Flowcell ID (or just leave the default)



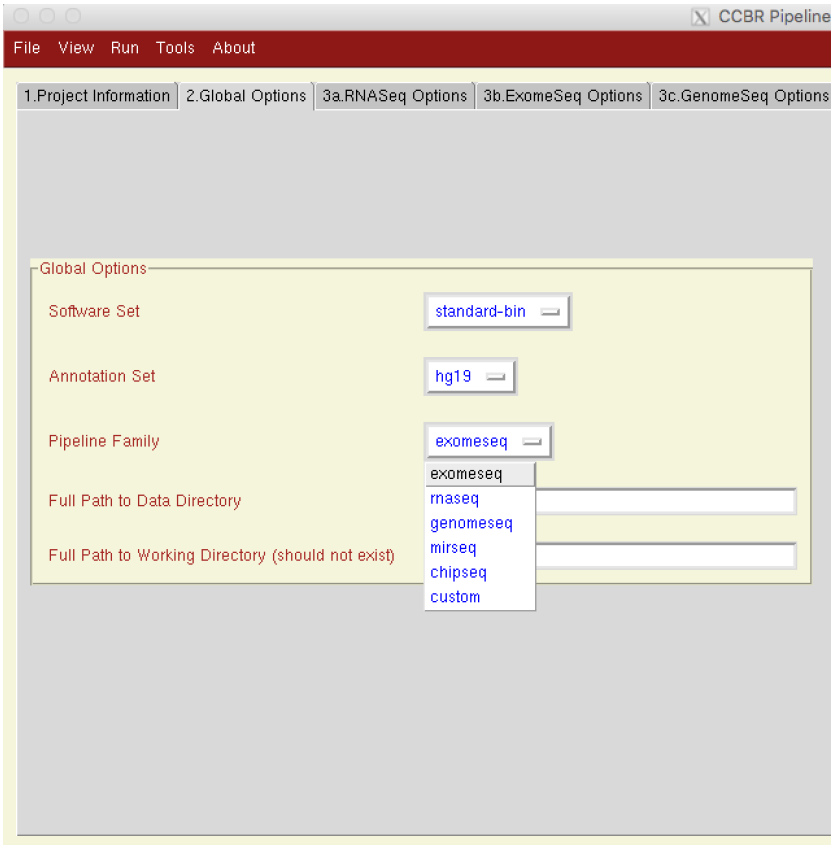
B2.3) Fill/Select global options (tab 2)



On this tab, you can specify the software set (only one set is currently available which is standard-bin), select the genome you wish to use under annotation set (i.e., select between mouse mm10 and human hg19):

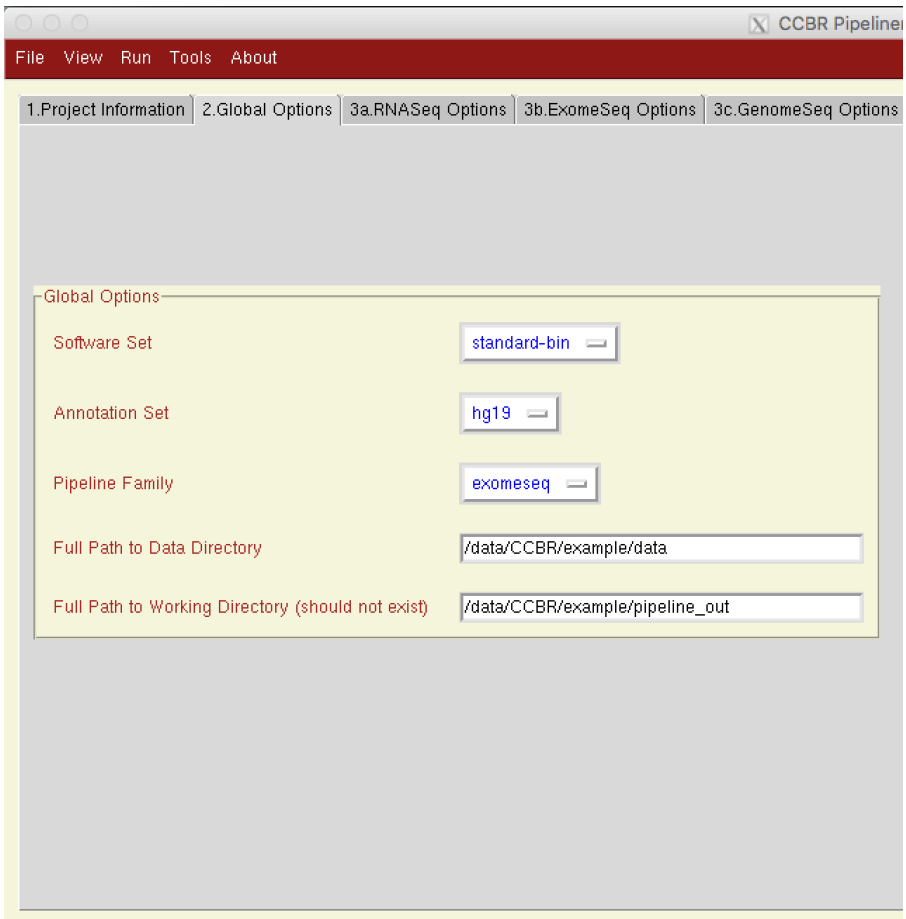


select the pipeline you wish to run:



Set the data source 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 Pipeliner, in which all Pipeliner files and result files will be generated:

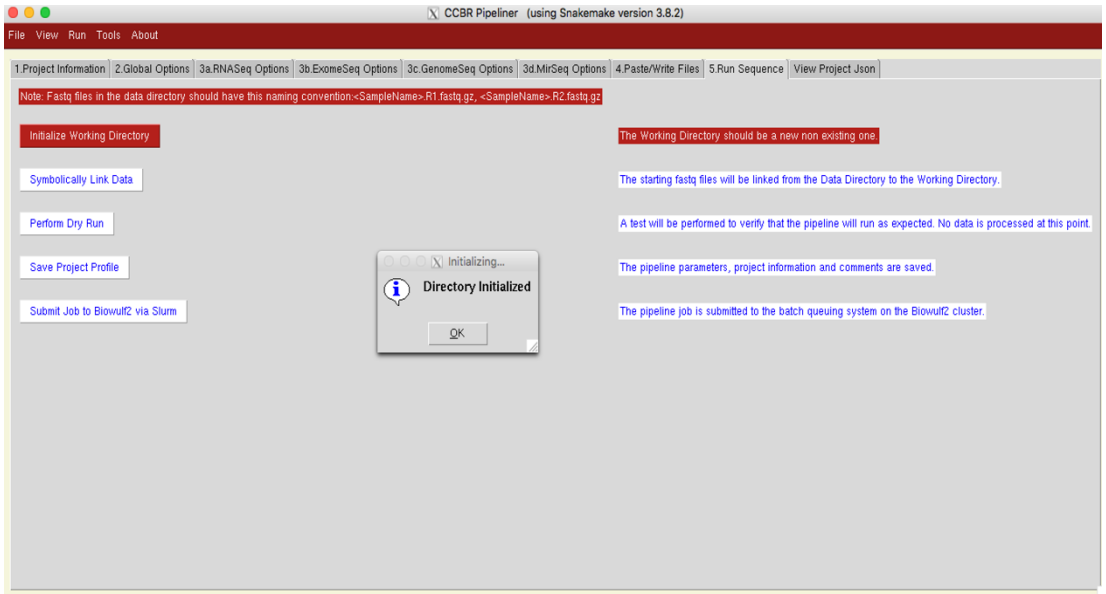


NB: You can fill the text boxes, or copy and paste (CTRL_V) the full path. You can also select the folders by using the Menu option file and choose data source or working directory.

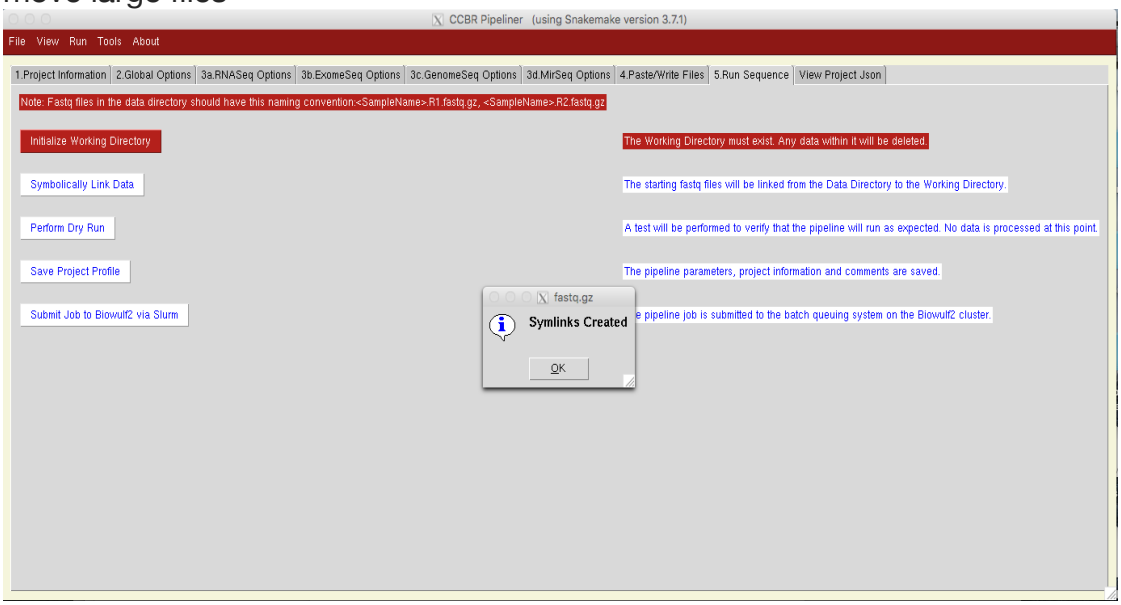
B2.4) Fill/Select pipeline-specific options on the specific pipeline tab (tab 3a-3c)

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

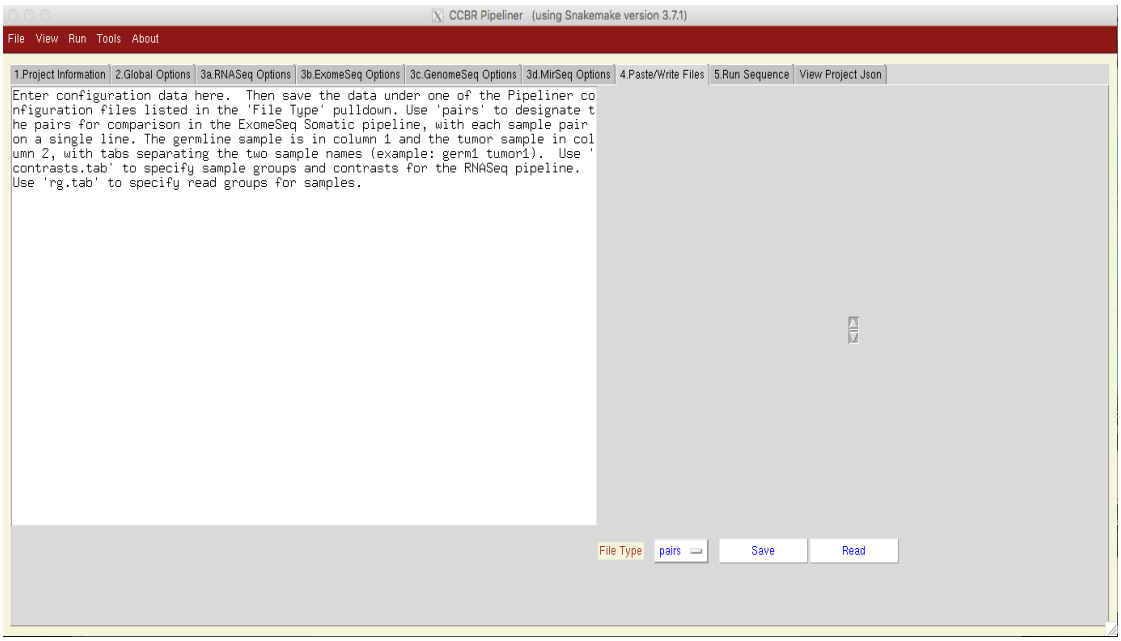
B2.5) Step through the Run Sequence (tab 5)
Initialize the working directory—Clicking this button clears the working directory, makes a few subdirectories within it, and populates these with a number of files needed to run the pipelines.



SymLink the data files to the working directory--this step creates symbolic links from the data files into the working directory to prevent the need to copy or move large files

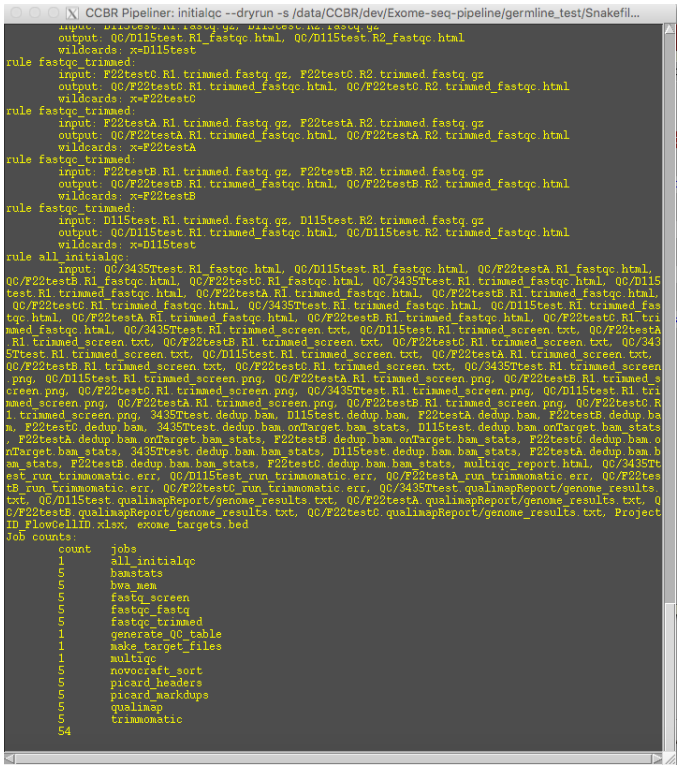


If additional files are required for your pipeline (see pipeline-specific instructions to determine if files in addition to read files are needed), move to Paste/Write Files tab (tab 4) and generate appropriate files:

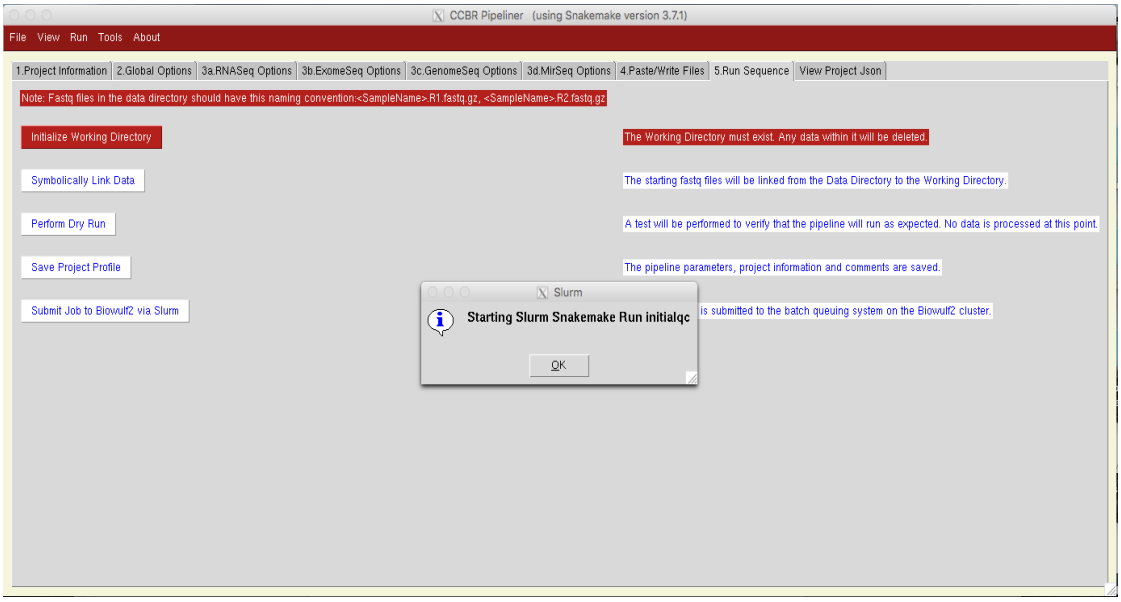


IMPORTANT!!! – these files need to be generated BEFORE running the dry run, or an error will occur.

B2.6) Return to the Run Sequence tab (tab 5). Perform a dry run to verify that the pipeline is configured properly—a successful dry run will look something like this:



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

Pipeliner 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

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

B6) Subdirectories used within working directory

- 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 (counts-based)

B8) Pipelines Planned for Inclusion

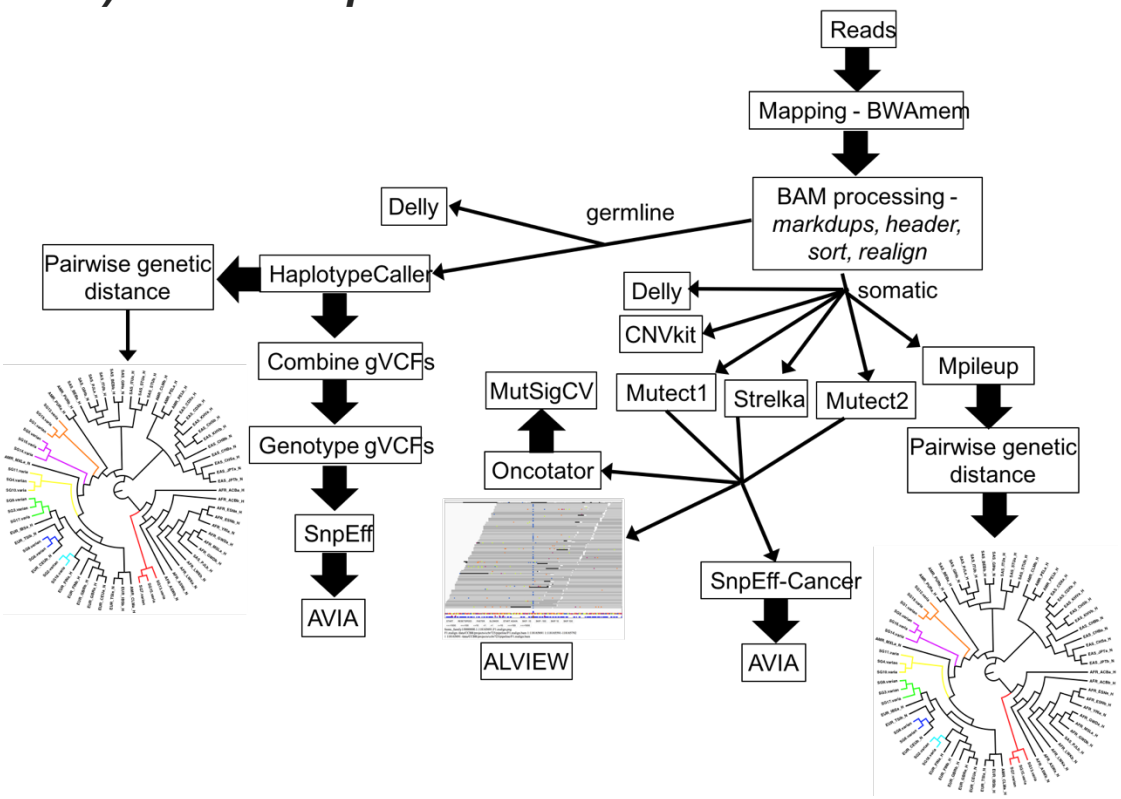
- ChipSeq
- mirSeq

B9) Organisms supported

- Human (hg19)
- Mouse (mm10)

C) Specific Instructions for executing the ExomeSeq/GenomeSeq pipelines

C1) Exome-seq



The Exome-seq pipeline can be run for either germline variant detection or paired tumor/normal somatic variant detection. Regardless of whether your goal is germline or somatic variant detection, the pipeline is run in two phases. First all reads must pass through the initalQC phase of read mapping, manipulation and filtering of BAM files, and QC assessment of read and mapping quality. So, beginning from fastq reads with the proper naming convention (Samplename.R1.fastq.gz, Samplename.R2.fastq.gz), you must select ‘initialqc’ from the Pipeline dropdown on the ExomeSeq Options tab:

In addition, you must point to the target bed file for the exome capture kit that was used to generate your sequence data. By default, the Agilent SureSelect v5+UTRs is used, but you can point to any other file you wish in the Target Capture Kit text box. If you do choose your own personal target bed file, there MUST be no header, chromosomes named WITHOUT the chr prefix (following the GRCh37 convention), and only the following 4 tab-delimited columns in this order:

Chromosome Start Stop Annotation

For initialQC, there are no additional files required, so you are now ready to proceed through the workflow on the Run Sequence tab, and submit your pipeline to biowulf.

Following completion of initialQC, deduplicated BAM files have been generated for all samples (Samplename.dedup.bam), and an overview of QC has been generated in the multiqc_report.html file.

QC Overview

The summarizing QC document is the multiqc_report.html file (generated with MultiQC; http://multiqc.info). This contains many interactive figures summarizing multiple aspects of your data, and can be viewed on any web browser. At the end of initalQC, overviews of read quality and map quality are presented:

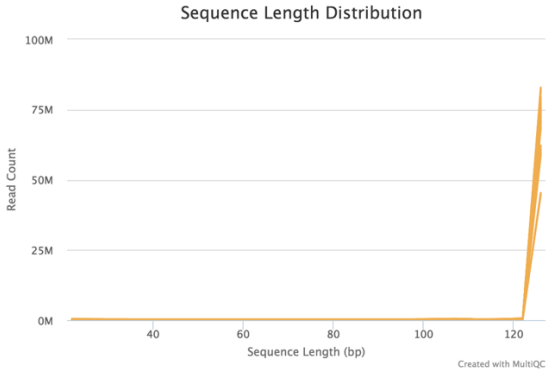
QC Summary Table:

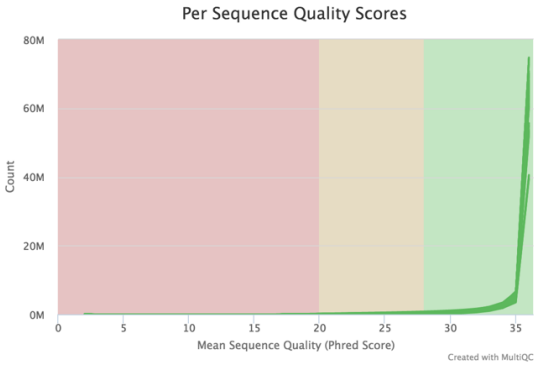
General Statistics

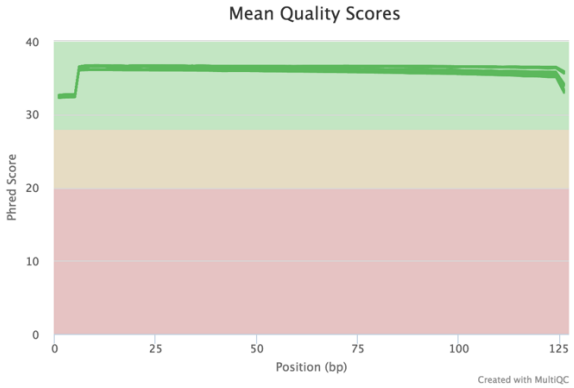
Copy table | Configure Columns | Showing 112/112 rows and 11/20 columns.

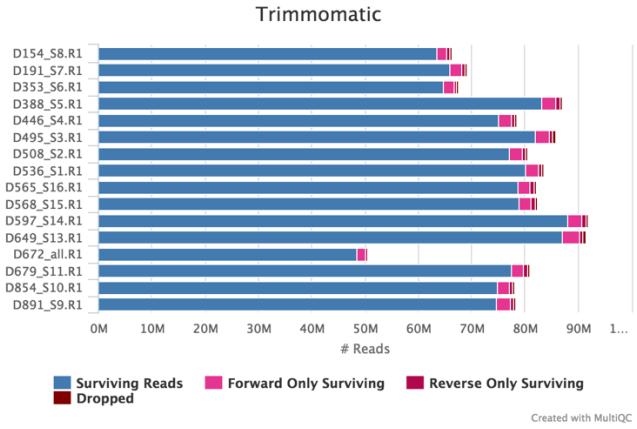
Sample Name	Avg. GC	Insert Size	≥ 30X	Coverage	% Aligned	% Dups	% Duplicates	% Mapped	% Dropped	% Dups	% GC	M Seqs
D154_S8.R1									0.6%	17.9%	46%	66.3
D154_S8.R1.trimmed										16.8%	46%	63.5
D154_S8.R2										18.1%	46%	66.3
D154_S8.R2.trimmed										17.0%	46%	63.5
D154_S8.dedup					99.1%		0.0%	100.0%				
D154_S8.qualimapReport	48%	191	95.3%	102.0								
D154_S8.sorted						5.2%						
D191_S7.R1									0.6%	18.1%	46%	69.1
D191_S7.R1.trimmed										17.0%	46%	66.0
D191_S7.R2										18.3%	46%	69.1
D191_S7.R2.trimmed										17.2%	46%	66.0
D191_S7.dedup					99.3%		0.0%	99.3%				
D191_S7.qualimapReport	47%	195	96.0%	104.0								
D191_S7.sorted						5.7%						

Read quality summaries:

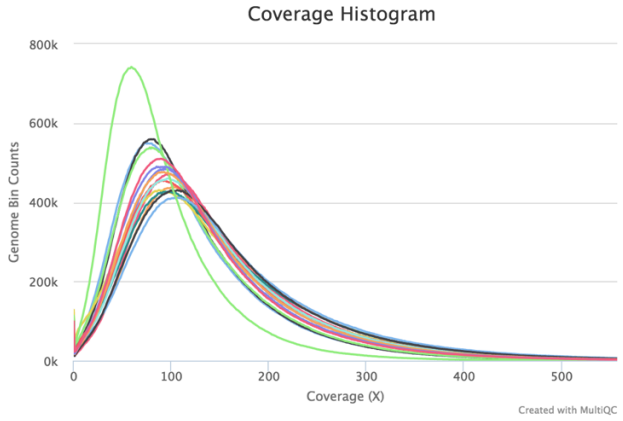


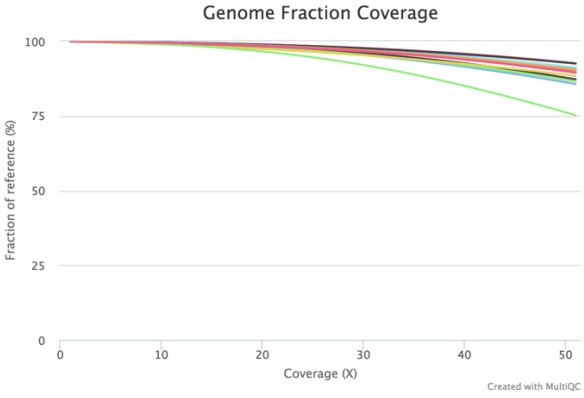




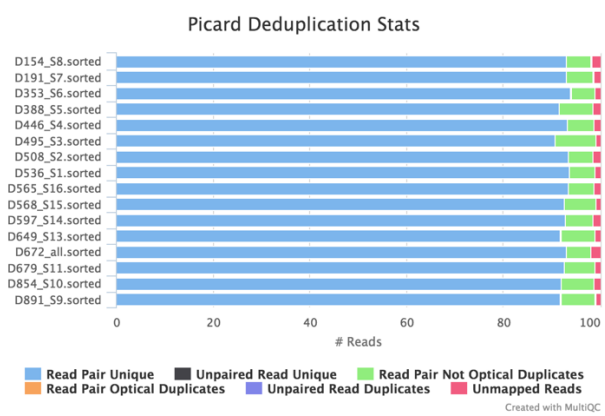


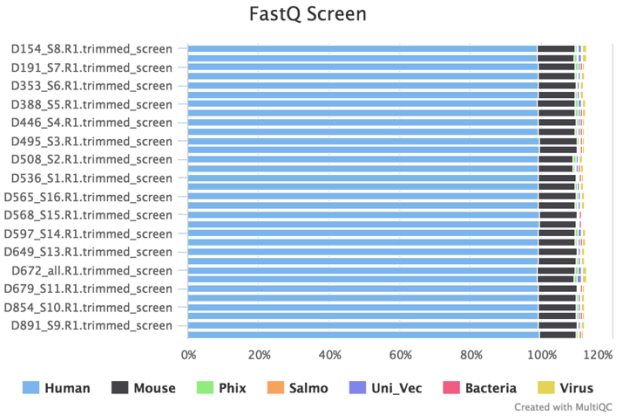
Qualimap Coverage Summaries:





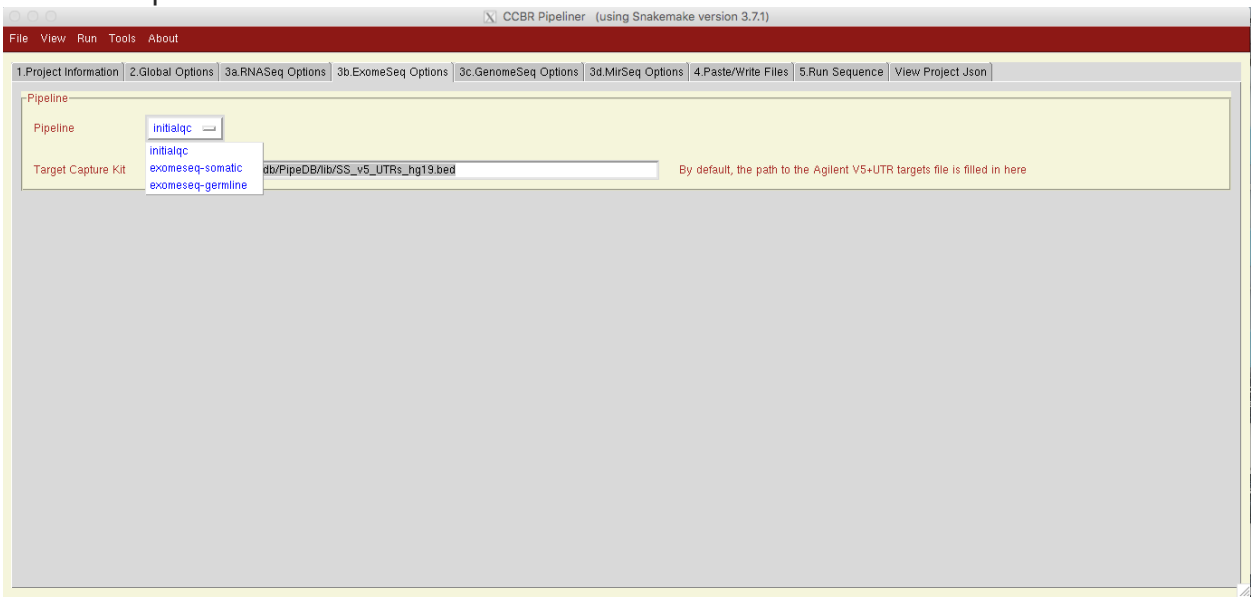
Optical duplicate and contamination summaries:





Variant Calling

To continue to variant detection, simply initiate the GUI as before, select ExomeSeq on the Global Options tab, navigate to the ExomeSeq Options tab, and select the desired ExomeSeq version:



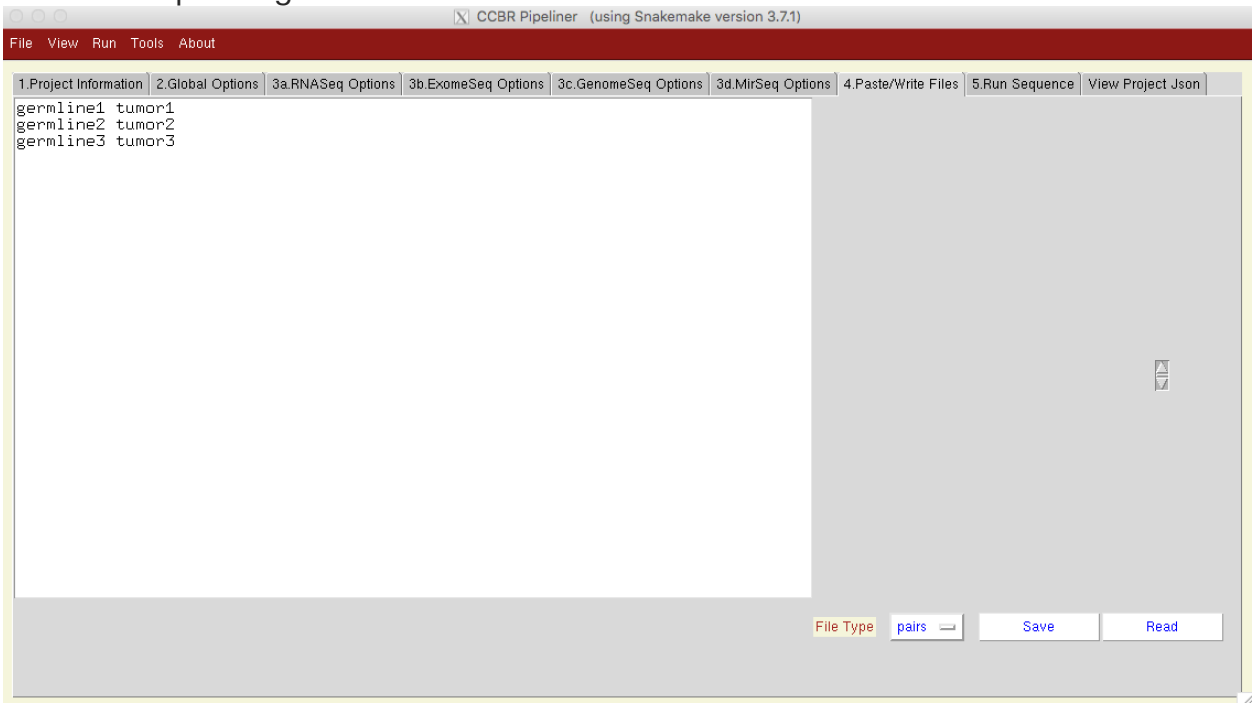
Then, ensure that you are pointing to the same Target Capture kit as was used in the initialQC phase.

For the germline ExomeSeq pipeline, no additional files are required, and you can proceed to the Run Sequence tab and start the pipeline. Do a dry run (skip initialization and symbolically linking data), and if the dry run completes, submit the job to biowulf.

IMPORTANT!!! – Following completion of the initialQC phase of the pipeline, **YOU NO LONGER NEED TO INITIALIZE THE WORKING DIRECTORY OR SYMBOLICALLY LINK DATA**. **If you mistakenly initialize, IT WILL DELETE ALL OF YOUR RESULTS.**

For the somatic ExomeSeq pipeline, you must generate the pairs file, which indicates the tumor/normal pairs for somatic variant calling. To do so, navigate to the

Paste/Write Files tab and paste in the pairs with each pair on a single line, the germline samplename in the first column, and the tumor samplename in the second column, with tabs separating columns. It should look like this:



Then, push the 'save' button, which will write the pairs file to the working directory. Then, you can proceed to the Run Sequence tab and start the pipeline. As above, do a dry run (******skip initialization and symbolically linking data******), and if the dry run recovers no errors, submit the job to biowulf.

C1.1) Results – ExomeSeq

Upon completion of the ExomeSeq pipeline (both somatic and germline), many resulting files are available for direct analysis, examination, and carrying forward to downstream application(s). Below is a list of the relevant files for the each pipeline. For further explanation of the outputs of various steps/programs, please refer to the tool-specific documentation:

C1.1.1) ExomeSeq germline:

- **<Samplename>.recal.bam** – for each sample, the final BAM alignment file from which variant detection was conducted
- **<Samplename>.g.vcf** – GVCF format variant file generated for each sample by the GATK Haplotype Caller (https://software.broadinstitute.org/gatk/).
- **combined.vcf** – joint-genotyping VCF generated by GATK Genotype gVCFs. This set of variants is filtered only on >=Q30, but NOT masked for exome targets, therefore including MANY off-target variants.
- **exome.recode.vcf** – final VCF, generated by masking combined.vcf for only variants within the exome target regions.
- **exome.snpeff.vcf** – SNPeff annotated version of the exome.recode.vcf

(http://snpeff.sourceforge.net)

- **snpeff.stats.html, snpeff.stats.csv** – summary outputs from SNPeff for the total
- **samples_and_knowns.vcf** – merged VCF resulting from combining exome.recode.vcf with a subset of the 1k genomes dataset consisting of 2 individuals from each of the 1k genomes populations (26 populations). This is a subset of loci, excluding those not found in the 1k genomes dataset.
- **sample_network.bmp** – image of phylogenetic network illustrating relationships among all analyzed samples, as well as the 1k genomes samples representing each of the populations included in 1k genomes dataset. This phylogenetic network is also generated in a crude text format in the 'outfile' file and in the newick tree format in the 'outtree' file. The outtree file is useful as input for phylogenetic tree visualization and analysis programs (i.e., FigTree)
- **full_annot.txt.zip** – large catalog of annotations generated by AVIA (http://avia.abcc.ncifcrf.gov). Annotations are in the same order as the variants in exome-recode.vcf, and can be opened with excel (after unzipping), allowing variants and annotations to be searched and filtered.
- **sample_vcfs** directory – directory containing a single VCF for each sample, extracted from exome.recode.vcf, as well as SNPeff outputs for each sample.
- **multiqc_report.html** – MultiQC (http://multiqc.info) summary html file capable of being opened in any web browser. This file contains interactive images summarizing read quality, alignment/mapping quality, and variant qualities by individual sample as well as combined across all samples.

C1.1.2) ExomeSeq somatic:

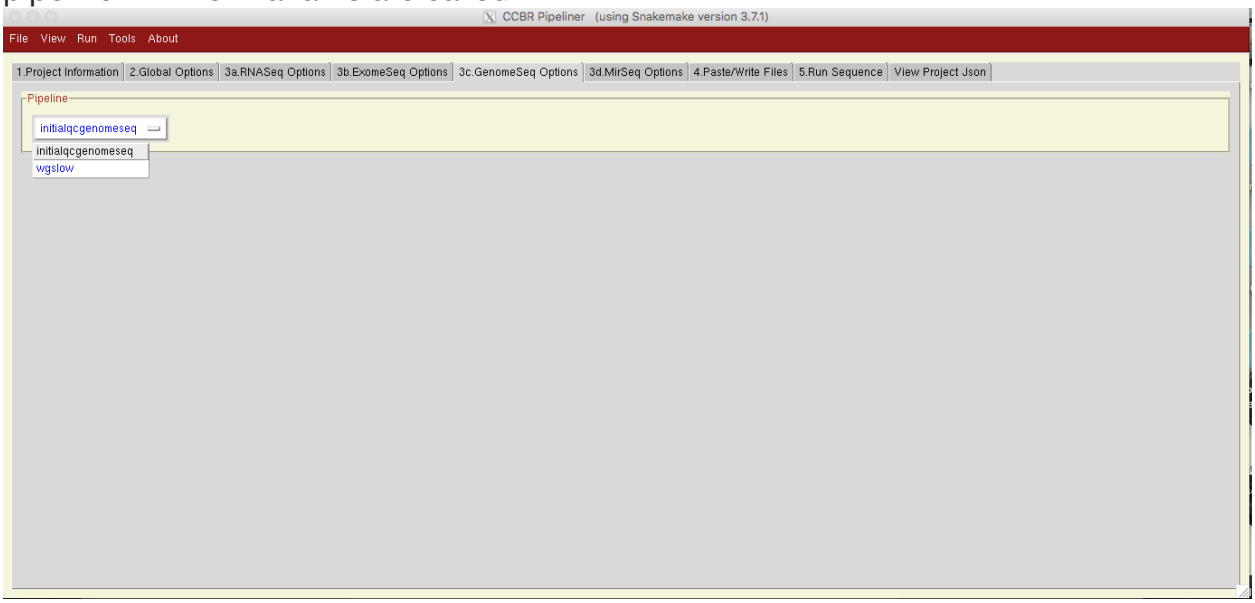
- **<Samplename>.recal.bam** – for each sample, the final BAM alignment file from which variant detection was conducted
- **strelka_out** directory – directory containing full set of results from Strelka (https://sites.google.com/site/strelkasomaticvariantcaller/home) somatic variant caller from all tumor/normal pairs indicated in 'pairs' file.
- **mutect_out** directory - directory containing full set of results from Mutect (http://archive.broadinstitute.org/cancer/cga/mutect) somatic variant caller from all tumor/normal pairs indicated in 'pairs' file.
- **oncotator_out** directory – within both the mutect_out and strelka_out directories, the oncotator_out directories contain the full annotated *.maf file for each tumor/normal pair generated by oncotator (http://portals.broadinstitute.org/oncotator/), as well as a concatenated mutect_variants_sorted.maf file containing all variants detected by mutect across all tumor/normal pairs.
- **mutsigCV_out** directory - within both the mutect_out and strelka_out directories, the mutsigCV_out directories contain the full set of outputs for the MutSigCV analysis (http://archive.broadinstitute.org/cancer/cga/mutsig).
- **snpeff.mutect.stats.html, snpeff.mutect.stats.csv, snpeff.strelka.stats.html,**

snpeff.strelka.stats.csv – summary outputs from SNPeff for mutect and strelka

- **germline_snps.vcf** – germline SNP calls for all samples (both tumor and normal)
- **samples_and_knowns.vcf** – merged VCF resulting from combining germline_snps.vcf with a subset of the 1k genomes dataset consisting of 2 individuals from each of the 1k genomes populations (26 populations). This is a subset of loci, excluding those not found in the 1k genomes dataset.
- **sample_network.bmp** – image of phylogenetic network illustrating relationships among all analyzed samples, as well as the 1k genomes samples representing each of the populations included in 1k genomes dataset. This phylogenetic network is also generated in a crude text format in the 'outfile' file and in the newick tree format in the 'outtree' file. The outtree file is useful as input for phylogenetic tree visualization and analysis programs (i.e., FigTree)
- **sample_network.bmp** – image of phylogenetic network illustrating relationships among all analyzed samples, as well as the 1k genomes samples representing each of the populations included in 1k genomes dataset. This output is useful in ensuring the tumor/normal pairs are correct. This phylogenetic network is also generated in a crude text format in the 'outfile' file and in the newick tree format in the 'outtree' file. The outtree file is useful as input for phylogenetic tree visualization and analysis programs (i.e., FigTree)
- **full_annot.txt.zip** – large catalog of annotations generated by AVIA for all somatic variations detected with the Mutect caller. Annotations are in the same order as the variants in merged_somatic.vcf.gz within the mutect_out directory, and can be opened with excel (after unzipping), allowing variants and annotations to be searched and filtered.
- **cnvkit_out** directory – full set of outputs for all tumor/normal pairs from the structural variant tool CNVkit
- **delly_out** directory – full set of outputs from the structural variant caller Delly, including calls for insertions (*ins.bcf), deletions (*del.bcf), inversions (*inv.vcf), translocations (*tra.bcf), and duplications (*dup.bcf).
- **mutect_variants_images.tar, strelka_variants_images.tar** – for both mutect and strelka, a tarred set of Alview (https://github.com/NCIP/alview) BAM images containing an html browsing file (i.e., mutect_variants_images.html). Opening this browsing file, you can then flip through images from the *recal.bam files of all variants occurring in genes that were mutated in at least 2 of your tumor/normal pairs. This unique set of outputs facilitates validation of somatic variants in genes mutated at least twice in your analysis. The images can be browsed with the right/left arrows on your keyboard, and contain annotations describing the sample they occur in, the number of samples the gene is mutated in, the position of the mutation, the gene it occurs in, and the context of the mutation (i.e., nonsynonymous, 3'UTR, etc.). Furthermore, the images are ordered Tumor1,Normal1,Tumor2,Normal2, etc. to facilitate rapid comparison of tumor/normal BAM files.

C2) GenomeSeq

The GenomeSeq pipeline is run exactly the same as the germline version of the ExomeSeq pipeline described above, with the initialqcgenomeseq portion of the pipeline being run first, followed by the wgslow portion, which is the portion of the pipeline in which variants are called.



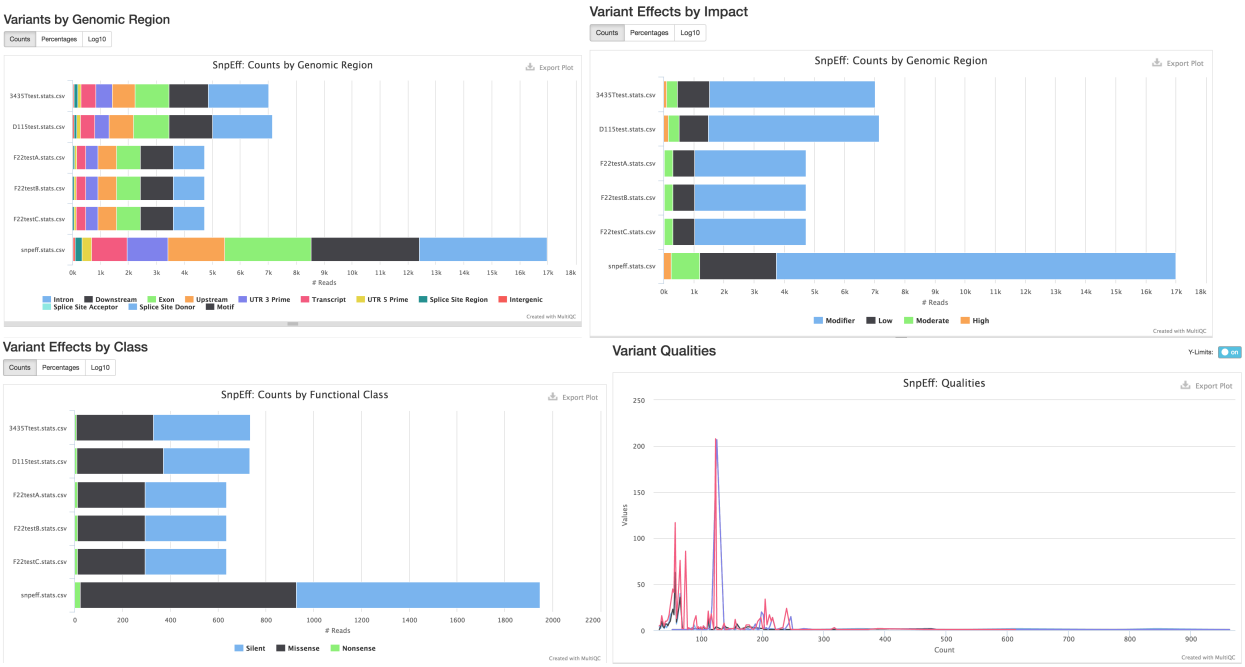
The only difference between the ExomeSeq and GenomeSeq workflows is that the GenomeSeq pipeline does NOT use a target bed file, as is specified in the ExomeSeq pipeline.

C2.1) Results – GenomeSeq germline

The files resulting from the GenomeSeq pipeline are largely identical to those of the ExomeSeq-germline pipeline. The only significant difference, is that combined.vcf is actually the final VCF file generated since no target masking is necessary for whole-genome sequencing. Therefore, all annotations cover all called variants passing the >=Q30 filter.

C2.1) Variant QC Summary

Following completion of your pipeline, MultiQC will do another full pass through your analysis, summarizing all of the QC metrics outlined above, and also add variant summaries to the multiqc_report.html document. So, in addition to the above plots, you'll get variant plots that summarize both the variants in each individual sample, as well as for the combined calls across all samples; it will look something like this:

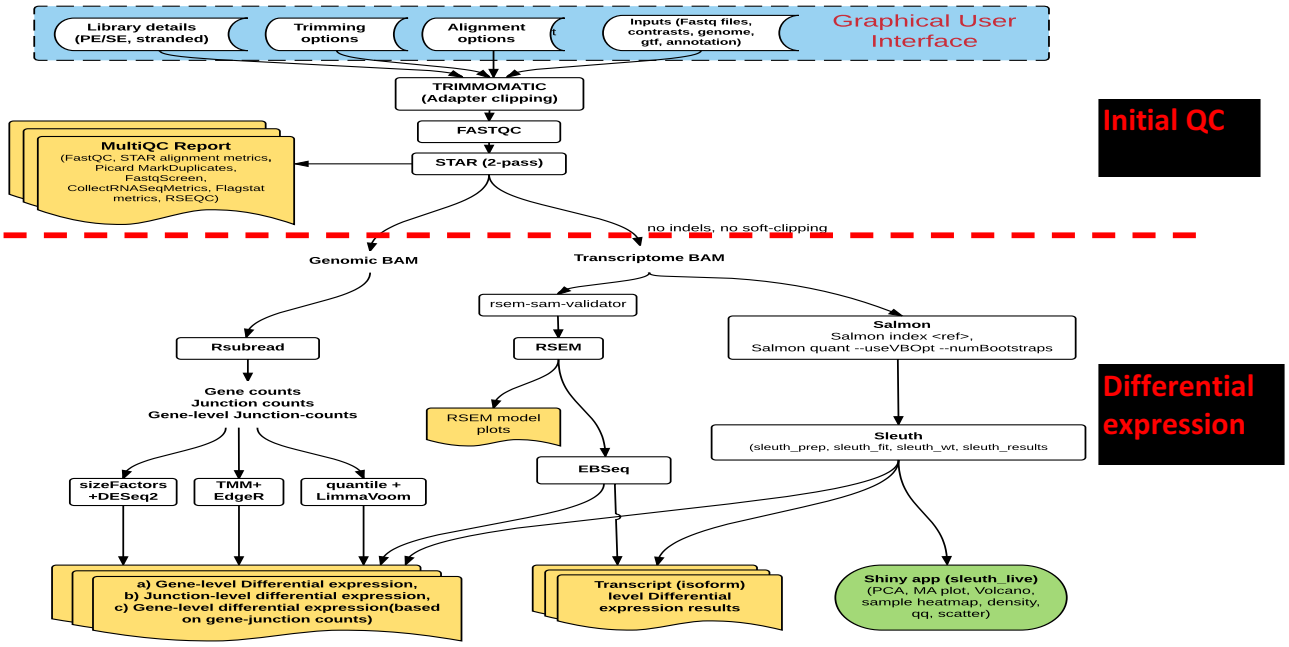


D) Specific Instructions for executing the RNASeq pipeline

D1) Introduction

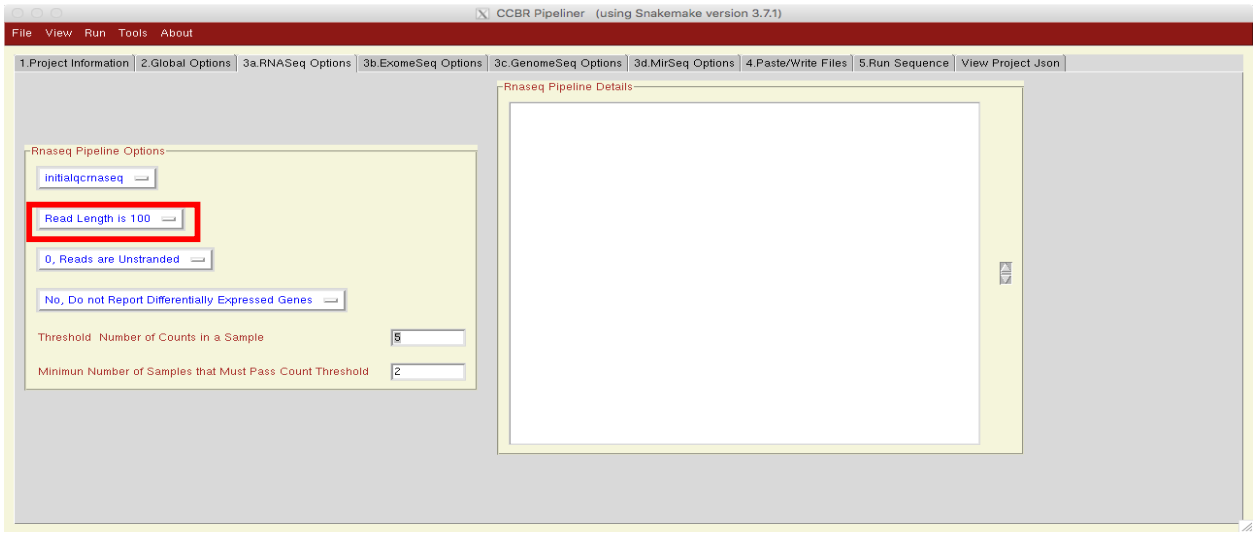
The RNASeq pipeline has been implemented to quantify gene & isoform expression and carry out tests for differential expression at both the gene- and isoform- level. The pipeline also generates results for differential expression across splice-junctions as well as differential expression at the gene-level, counting only junction-spanning reads.

The RNASeq pipeline is run in two phases. First all reads must pass through the initalQC phase of adapter-clipping, read-level QC, read-alignment and alignment-level.



D2) InitialQC

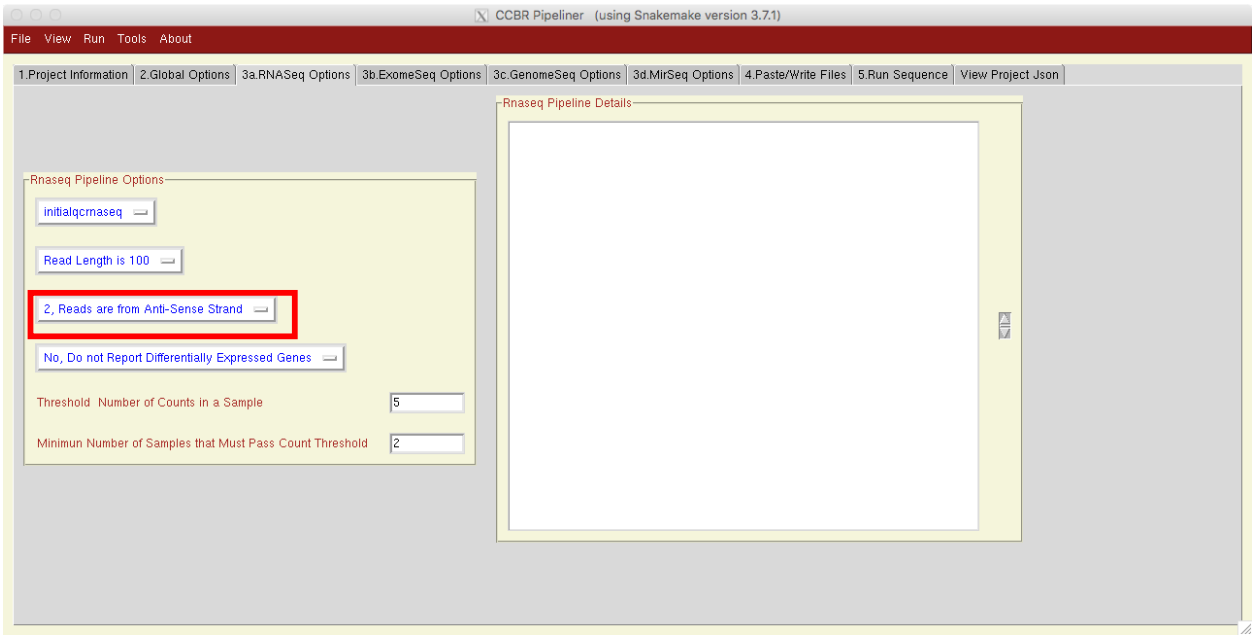
After entering information on the 'Project Information' (Section B2.2) and 'Global Options' tab (Section B2.3), you click on the '3a.RNASeq Options' tab. Now select the RNASeq Pipeline Options as follows:
D2.1) First, select initialqcrnaseq as shown below:



D2.2) Select the read-length for the fastq reads in your dataset (options are 50, 75, 100, 125, 150 and 250bp). If the read-length does not match any of the options in the

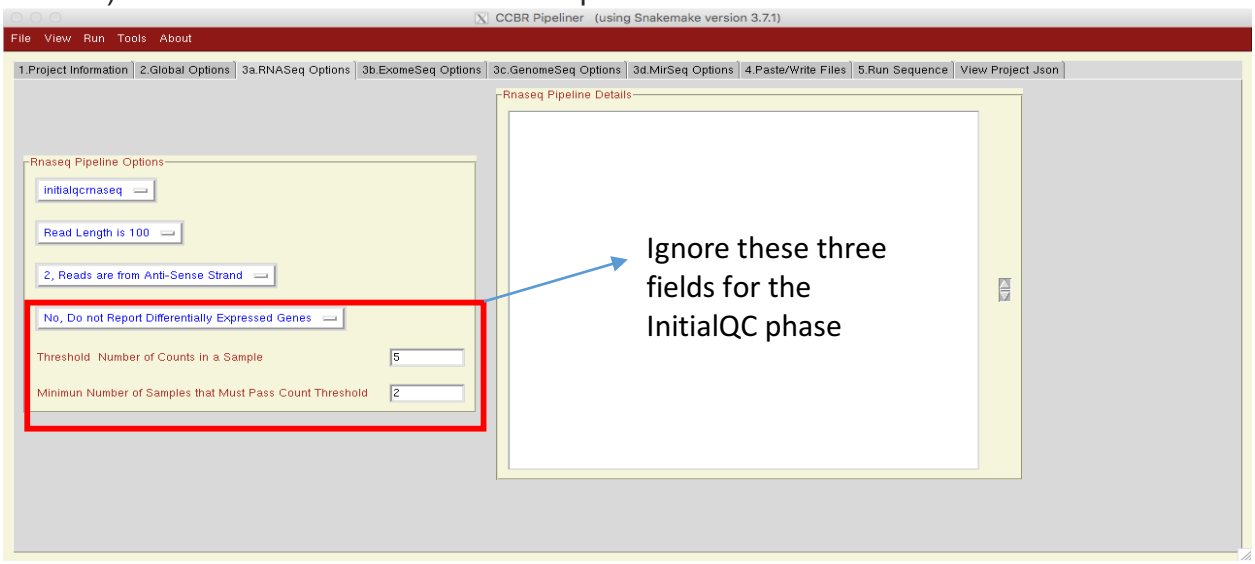
drop-down, select the length that is closest to the actual read-length

D2.3) Select the strandedness of your reads. Options are: Unstranded, Sense Strand or Anti-Sense Strand. Typically, if your sequencing libraries are generated using Illumina TruSeq library prep protocols, you would select 'Anti-Sense Strand' indicating that Read1 maps to the anti-sense strand. For Nextera library prep, 'Unstranded' is selected.



D2.4) The next three input fields are not used for the InitialQC phase, so you can simply ignore these.

D2.5) Now click on the tab '4.Run Sequence' and follow the instructions in section



D2.5 of this document to initialize the working directory, create symbolic links to the fastq files, test a dry-run and submit the pipeline execution to the Biowulf cluster.
NOTE: Once the pipeline job is submitted, you can close the GUI. On the command prompt, you can check the status of your jobs by using this command:

- squeue –u <username>

D2.6) Results from InitialQC

Once the pipeline completes successfully, the working directories will contain the following folders/files:

- **QC folder**: contains Fastqc results on the adapter-clipped fastq files
- **trim folder**: contains adapter-clipped fastq files (both paired and unpaired)
- **STAR 1st pass output files**:
  *.Aligned.out.bam
  *.SJ.out.tab
  *.Log.progress.out
  *.Log.final.out
- **STAR 2nd pass output files:**
  *.p2.Aligned.toTranscriptome.out.bam
  *.p2.ReadsPerGene.out.tab
  *.p2.Aligned.sortedByCoord.out.bam
  *.p2.SJ.out.tab
  *.p2.Log.progress.out
  *.p2.Log.final.out
  *.star_rg_added.sorted.bam
  *.star_rg_added.sorted.dmark.bam
  *.star_rg_added.sorted.dmark.bai

- **Alignment QC files:**
  a) *.RnaSeqMetrics.txt: Output from Picard CollectRNASeq Metrics
  b) *.flagstat.concord.txt: Output from samtools flagstat
  c) RSEQC output:
  *.inner_distance_plot.pdf
  *.inner_distance_plot.r
  *.inner_distance_freq.txt
  *.inner_distance.txt
  *.GC.xls
  *.GC_plot.pdf
  *.strand.info
  *.Rdist.info

  d) ProjectID_FlowCellID_Summary.txt : comprehensive Alignment QC metrics from RSEQC
  e) ProjectID_FlowCellID.xlsx: QC-metrics and barcharts in excel format

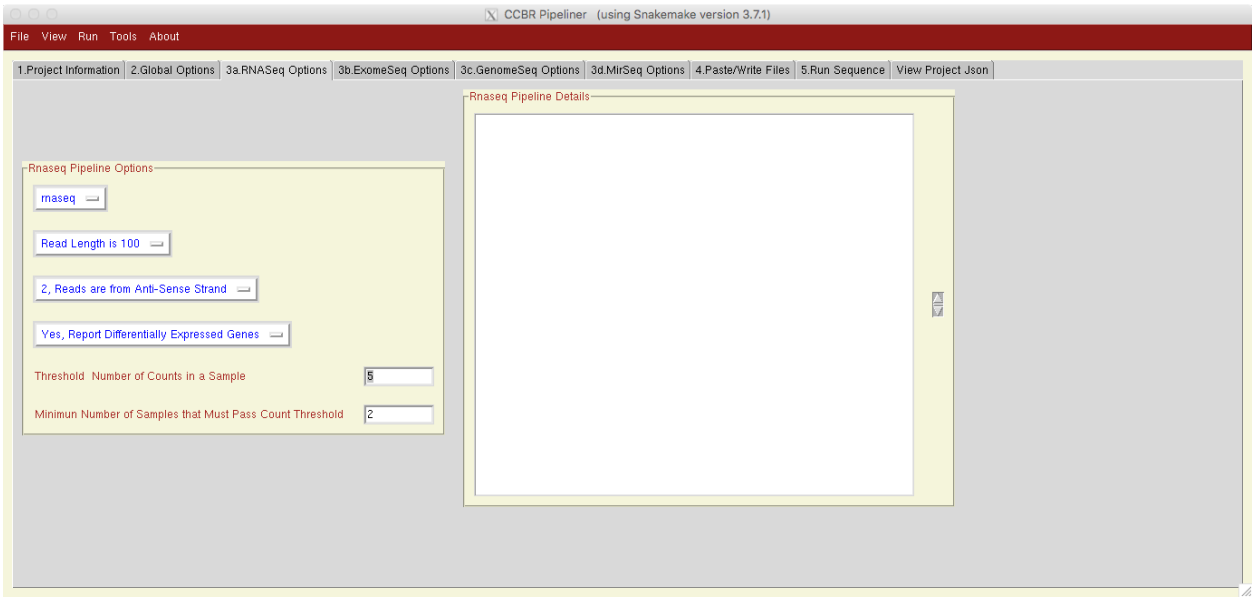
f) FQscreen folder: contains results from Fastq-Screen

- **Reports folder:**
  a) aggregate_fastqc_report.html: quick view of PASS/WARN/FAIL flags using color codes, across all samples and all metrics
  b) multiqc_report.html: comprehensive and interactive HTML page aggregating QC metrics across several tools

D3) Differential expression workflow (after InitialQC)

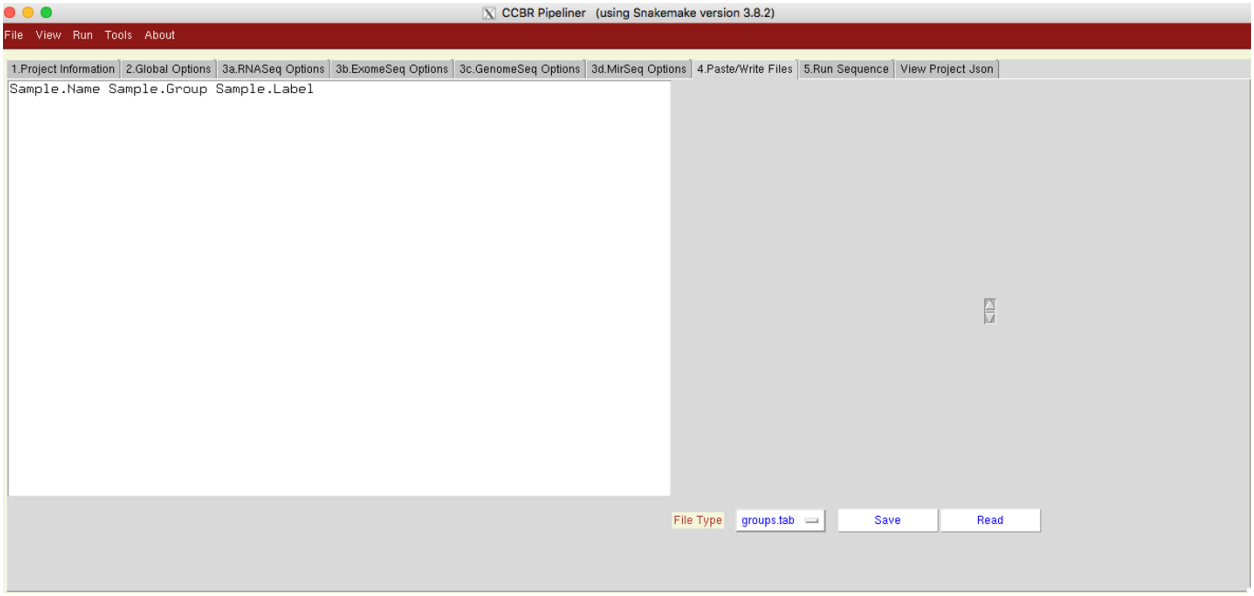
This workflow is to be executed in the same working directory as that specified in the InitialQC phase, so that the pipeline can identify the output files from the InitialQC run, that are being used as input files to the differential expression workflow.

D3.1) Following the same commands as used in section B2.1-2.3, launch the Pipeliner GUI, fill in Project Information and Global Options. Then on the tab '3a.RNASeq Options', choose 'RNASeq' pipeline, choose the read length, specify the read strandedness and choose whether to report differentially expressed genes or not. Next, the user has to fill in two more text boxes to specify the gene-filtering criteria. For a gene to be included for differential expression, it has to have a minimum number of counts per sample, across a minimum number of samples. By default, we choose a threshold of 5 counts in a minimum of two samples. This filtering removes genes that are not expressed in any sample (all counts are zeros), as well as filters out genes with very low counts below the chosen count threshold.



D3.2) Next, navigate to the '4.Paste/Write Files' tab to specify the sample names, their group assignments, short labels to identify samples on various plots and specify the contrasts of interest.

On the bottom right, the ‘File type’ dropdown lets a user specify information pertaining to that file (either pairs, rg.tab, groups.tab or contrasts.tab). The RNASeq pipeline uses two files: the groups.tab and contrasts.tab. So first choose groups.tab on this dropdown and then choose contrasts.tab if you would like to have the list of differentiallially expressed for your contrasts. When you choose a file, the big text box on the left is filled in with one header line indicating the order of information to be provided column-wise.

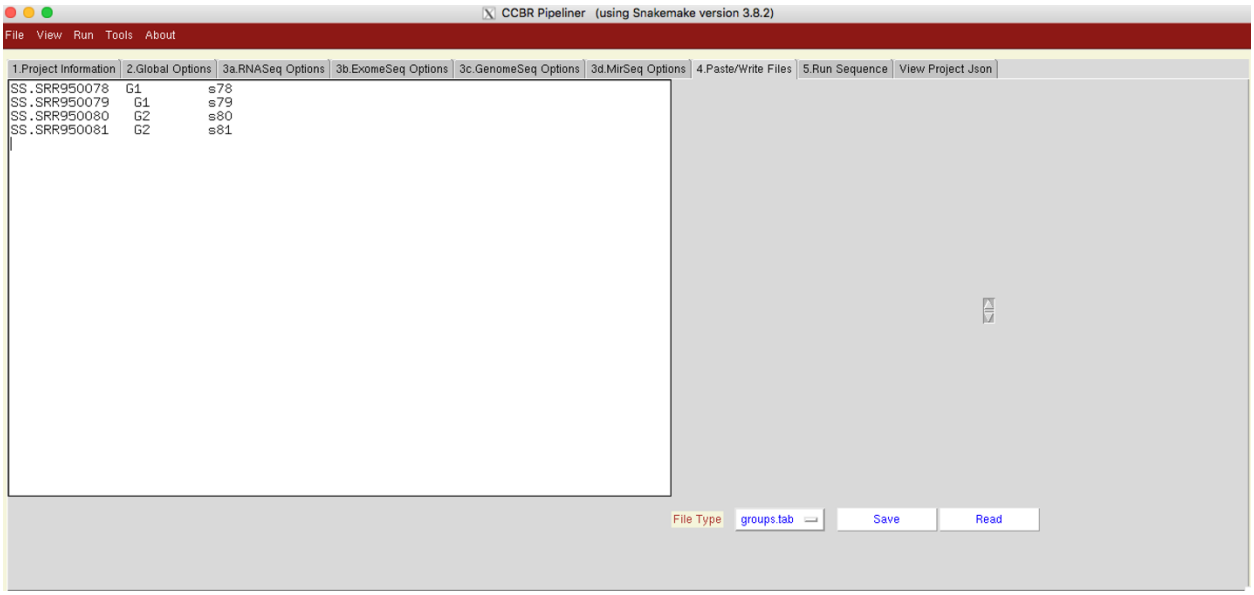


D3.3) The groups.tab to be in columns with 3 columns in order: samplename, samplegroup and samplelabel.

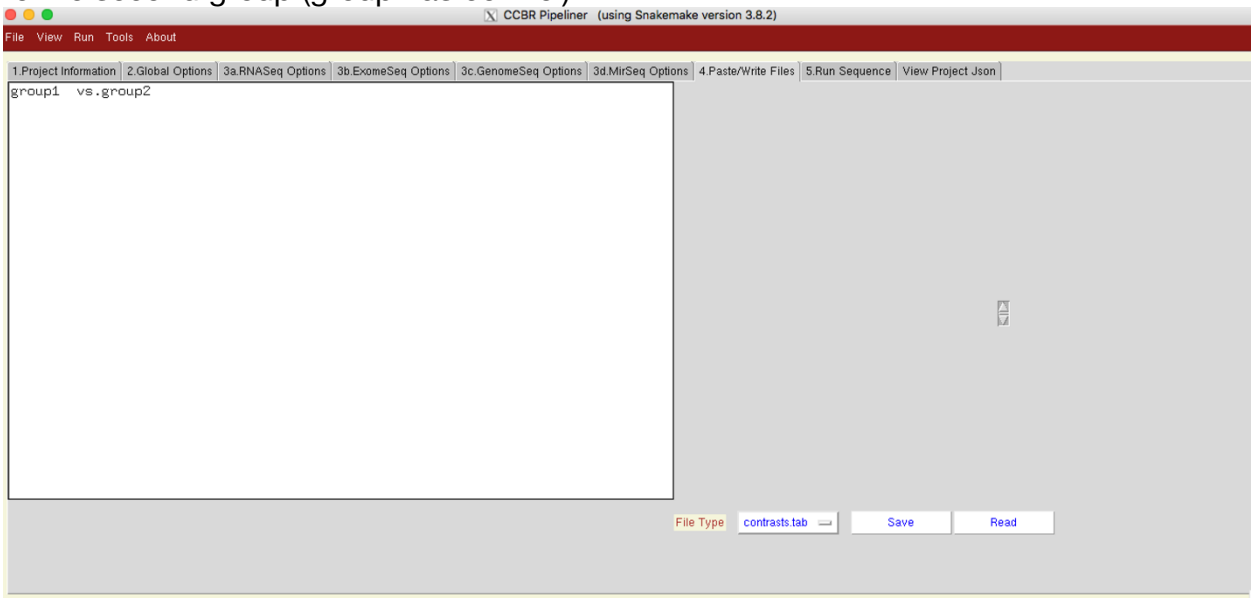
Here one example:

SS.SRR950078	G1	s78
SS.SRR950079	G1	s79
SS.SRR950080	G2	s80
SS.SRR950081	G2	s81

You can either type into this text box, or if you have a large number of samples, you can simply create this tabular information in an excel spreadsheet, save the spreadsheet as tab-delimited text file, and copy the information and paste it into the text-box. If you choose to copy and paste from a file, please ensure there are no extra lines or carriage returns after the last row of information. Also, the default header row has to be deleted. **After filling in the information, make sure to click on ‘Save’ to save this information in the groups.tab file**. Here is a snapshot of how this text-box would look after filling in the information:



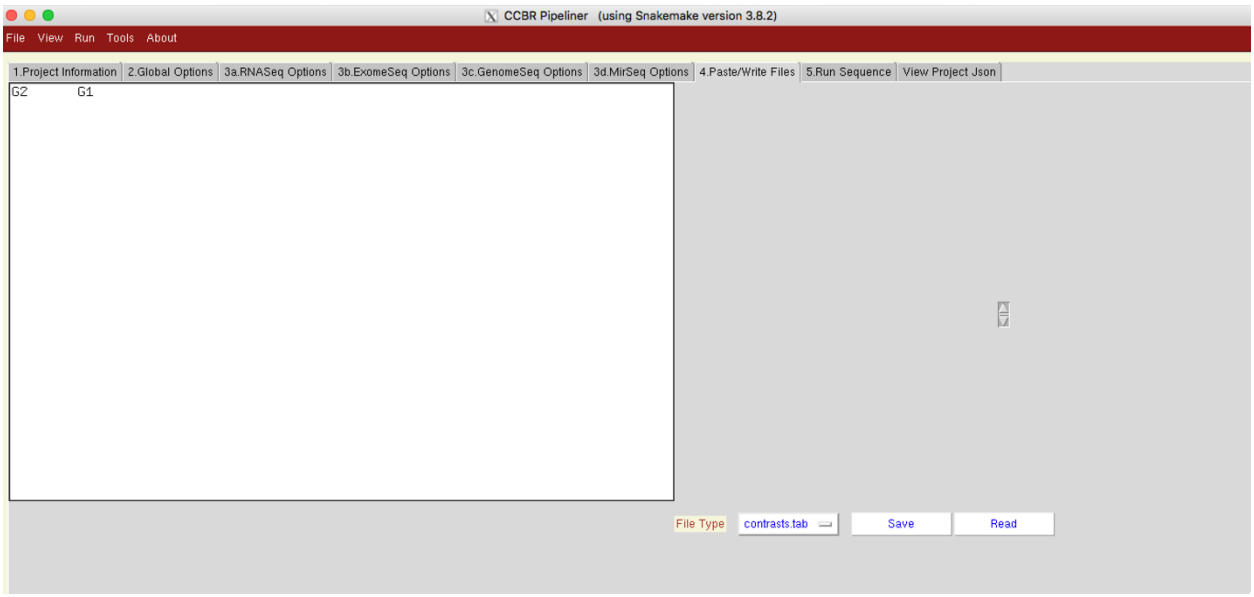
D3.4) The contrasts.tab to be in columns with 2 columns in order: group1 and group2. You need to specify a contrast per line where the first group (group1) will be compared to the second group (group2 as control).



Here one example:

G2 G1

After filling in the information, make sure to click on 'Save' to save this information in the contrasts.tab file. Here is a snapshot of how this text-box would look after filling in the information:



D3.5) Now navigate to the '5. Run Sequence' tab, perform a Dry-run and submit the pipeline for execution.

D3.6) Results from RNASeq pipeline

Once the pipeline completes successfully, the working directory should contain the following files/folders:

- **Folder salmonrun**: Contains one folder for each sample, containing salmon quantification files
- **Folder DEG_genes**: contains results of differential expression at the gene-level, from three DE algorithms: Limma, DESeq2 and EdgeR
- **Folder DEG_geneJunctions**: contains results of differential expression at the gene level, counting only exon-exon junction-spanning reads (useful if the library is total RNA, where we see a large proportion of intronic reads, originating from unspliced RNA). All 3 DE methods are applied to find differentially expressed genes.
- **Folder DEG_junctions**: contains results of differential expression at the junction-level.Here we are simply quantifying reads across each known splice-junction in the chosen genome. So the output contains read counts for each exon-exon junction in the genome (not collapsing by gene). Then all 3 methods are applied to find differentially expressed junctions.
- **RSEM results**:
  *.rsem.genes.results: gene-level counts from RSEM
  *.rsem.isoforms.results: isoform-level counts from RSEM
- **EBSeq results**:
  *.ebseq: differential expression results from EBSeq at the isoform- level