RNA-Seq Processing Workflow V.1

cameron_baker [1]

[1]University of Rochester

Version 1

Sep 15, 2021

In Development Share

This protocol is published without a DOI.

cameron_baker

ABSTRACT

This protocol contains details for running the RNA-Seq processing workflow as well as steps detailing underlying programs used and evaluation criteria for overall performance of the workflow.

PROTOCOL CITATION

cameron_baker 2021. RNA-Seq Processing Workflow. **protocols.io**
https://protocols.io/view/rna-seq-processing-workflow-bx9xpr7n
Version created by cameron_baker

CREATED

Sep 15, 2021

LAST MODIFIED

Sep 15, 2021

PROTOCOL INTEGER ID

53271

Running the workflow

1 ⚠

Running the workflow consists of a single step, kicking off the run command with the required parameters. The parameters and how to correctly set them will be described here, but the internal run commands and reasonings behind them will be explained below.

We start by cloning the repository containing the workflow into a given project folder.

Clone Repository

```
git clone https://gitlab.circ.rochester.edu/GRC/RNA-Seq_processing_v3.git
```

Before we launch the workflow, we need to edit the muliqc configuration file so that the final report has the correct fields reported. Edit lines 15, 16, and 20 with information located within the Trello card for the project.

```
vim RNA-Seq_processing_v3/.multiqc_config.yaml
```

Then, from that project folder, we launch the workflow.

Launch Workflow

./RNA-Seq_processing_v3/initiateRnaSeq REFERENCE STRANDED SAMPLES

The parameters to pass in are the REFERENCE, a number indicating whether or not the samples are STRANDED, and a list of SAMPLES.

REFERENCE - A full path the the desired reference

The full path to the most up to date mouse and human references within our ecosystem are as follows:

```
/scratch/grc_group/genomeMapping/mg38/M25/
/scratch/grc_group/genomeMapping/hg38/gencode36/
```

These are taken from gencode. Reference build instructions will be detailed elsewhere.

STRANDED - nature of the library preparation related to transcription orientation

This parameter is either 0 or 2, depending on whether or not the library preparation associated with the project is unstranded or not. **The only RNA-Seq libraries that are unstranded are the Ultra-low input RNA-seq kits**. This information is contained within the Trello card associated with the project. All other projects have a STRANDED parameter of 2.

The reason why it is reverse stranded (see featureCounts step) as opposed to stranded is that the Illumina TruSeq Stranded mRNA library preparation kit, the second read dictates the strandedness of the read. More information related to this is contained within this biostars post.

SAMPLES - the list of samples to be processed

This is usually Sample_* to run all samples within a given project, but the workflow may be run on any subset of the samples, passed in accordingly.

Data Cleaning

2 We use fastp to both trim and provide basic QC reports for further aggregation within multiqc. The general run command is included below. There are various differences across paired-end and single end libraries, but the major on is that paired end libraries samples are prepended with "synced_" relative to single end libraries.

fastp Run Command

```
/bin/time -v fastp \
                --in1 ${SAMPLE}_R1.fastq.gz \
                --out1 clt_${SAMPLE}_R1.fastq.gz \
                --length_required 35 \
                --cut_front_window_size 1 \
                --cut_front_mean_quality 13 \
                --cut_front \
                --cut_tail_window_size 1 \
                --cut_tail_mean_quality 13 \
                --cut_tail \
  --adapter_fasta ../RNA-Seq_processing_v3/adapters.fa \
                -w ${CPUS} -y -r \
                -j ${SAMPLE}_fastp.json
```

Each of the run commands begins with the following

```
/bin/time -v
```

This allows us to capture memory and computation usage for each run command, and is recorded in the logs specific for that command.

--length_required: the minimum length to keep a given read
--cut_front_window_size: this trims the beginning of the read for a specific quality
--cut_front_mean_quality: the quality score required to keep a base at the beginning of a read
--cut_tail_window_size: this trims the end of the read for a specific quality
--cut_tail_mean_quality: the quality score required to keep a base at the end of a read
--cut_front and --cut_tail indicate that we want to perform the sliding window quality filter
--adapter_fasta: a predefined list of adapters to trim from the reads
-w: the number of CPUs to run with fastp
-y: a filter for reads of low complexity
-r: similar to the cut parameters above. This is a sliding window quality filter across the whole read with a minimum quality of 20 and a window size of 4
-j: outputs a json report summarizing the run

Sequence Alignment

3 STAR Alignment

We use STAR as the classic alignment tool to arrive at aligned reads (as bam's) for further usage within read quantification.

STAR Alignment

```
/bin/time -v STAR \
                --twopassMode Basic \
                --readFilesCommand zcat \
                --runThreadN ${CPUS} \
                --runMode alignReads \
                --genomeDir ${STARREF} \
                --readFilesIn clt_${SAMPLE}_R1.fastq.gz \
                --outSAMtype BAM Unsorted \
                --outSAMstrandField intronMotif \
                --outFileNamePrefix clt_${SAMPLE}_ \
                --outTmpDir ./tmp/star \
                --outFilterIntronMotifs RemoveNoncanonical \
                --outReadsUnmapped Fastx ;
```

--twopassMode: performs two-pass alignment on the sample. Two-pass alignment has been shown to improve splice junction quantification
--readFilesCommand: uses zcat (cat on gz compressed data) as the method for reading in the data
--runThreadN: the number of CPUs allowed for the program
--runMode: us telling the program that we want to perform read alignment against a given reference
--readFilesIn: the file to be processed
--outSAMtype: the output format of the aligned reads
--outSAMstrandField: similar to the strand field within cufflinks. Filters out reads with inconsistent or non-canonical introns
--outFileNamePrefix: Prefix for the outfile
--outTmpDir: temporary directory for STAR files
--outFilterIntronMotifs: Filters out alignments containing non-canonical junctions
--outReadsUnmapped: output format of the unmapped reads

Note: when rerunning sample sets, you will need to delete Sample_*/tmp or STAR will fail to run

4 Sambamba for alignment sorting

We use Sambamba to sort the output alignments from STAR. Sorted reads are required for downstream processes.

Sambamba sort

```
/bin/time -v sambamba sort \
                -t ${CPUS} \
                -m 2G \
                --tmpdir='./tmp' \
                -o clt_${SAMPLE}_R1.bam \
                clt_${SAMPLE}_Aligned.out.bam ;
```

-t: the number of CPUs allowed for the process
-m: the amount of RAM allowed for the process
-o: the output file name

Read Quantification

5 Subread featureCounts

featureCounts is used as the main read counting tool within the RNA-Seq workflow.

featureCounts

```
/bin/time -v featureCounts \
                -s ${STRANDED} \
                -T ${CPUS} \
                -t exon \
                -g gene_name \
                -a ${STARREF}genes.gtf \
                -o ${SAMPLE}_featCountsU.txt \
                clt_${SAMPLE}_Aligned.out.bam \
                &> ${SAMPLE}_countU.log ;
          cut -f1,7 ${SAMPLE}_featCountsU.txt | \
                tail -n +3 > ${SAMPLE}_countsU.txt ;
```

-s: the strandedness of the samples
-T: the number of CPUs designated for the task
-t: map at an exon level
-g: while it maps at an exon level, we count each exon map as the main reported feature we map to
-a: the path to the transcriptome reference for mapping
-o: the name of the output file

The final step of the command creates a gene/count tab separated data file for downstream processing.

5.1 Salmon Pseudo-alignment

Salmon is run in conjunction with STAR with the goal of mapping to transcript as opposed to mapping gene. We only perform secondary analysis on human and mouse projects. More information related to salmon quant is located here. Descriptions of run parameters are taken from the documents verbatim. Salmon serves as an alignment and read quantification tool. Salmon also serves as an alternative to featureCounts for quantifying multi-mapping reads.

salmon

```
/bin/time -v salmon quant \
                -i ${STARREF}/salmon/ \
                -l A \
                --seqBias \
                --gcBias \
                --posBias \
                -r clt_${SAMPLE}_R1.fastq.gz \
                -p ${CPUS} \
                -o salmon_${SAMPLE} \
                 &> ${SAMPLE}_salmon.log  ;
```

-i: the full path to the salmon reference. The Salmon references live the REFERENCE path.
-l A: allows Salmon to automatically infer the library type
--seqBias: enables Salmon to learn and correct for sequence-specific in the input data
--gcBias: enables Salmon to attempt to correct for biases in how likely a sequence is to be observed based on its internal QC content
--posBias: enables modeling of a position-specific fragment start distribution. This is meant to model non-uniform coverage biases that are sometimes present in RNA-seq data (e.g. 5' or 3' positional bias)
-r: the input read file. This is replaced with -1 and -2 for paired end read files
-p: the number of cores used
-o: the name of the output folder

Contamination Detection

6 Centrifuge is used to detect presence of sequence not mapping to reference correspond to the samples. Upon further examination of the final multiqc reports, we may fall back to centrifuge with hopes of finding sources of poor alignment rates. These reports are not sent to investigators unless otherwise requested or directed. Below is a snippet of the centrifuge run command located within centrifuge.sb. All centrifuge results are tabulated for a given project, reporting the top matching taxa on a sample to sample basis.

Centrifuge

```
/bin/time -v centrifuge \
    -p ${CPUS} \
    -x ${CENTRIFUGE_DB} \
    -U ${INPUT_FASTQ1} \
    -S ${CENTRIFUGE_RESULTS} \
    --report-file ${CENTRIFUGE_REPORT}
```

-p: number of CPUs allowed for the process
-x: the centrifuge database used for sample mapping
-U: the input fastq. Replaced with -1 and -2 for paired end sample sets
-S: the output file name for classification results
--report-file: the classification summary results

Workflow Evaluation

7 ⚠

We use Multiqc to evaluate the success of the RNA-Seq workflow. See the report below for an example report. These reports accompany the final delivery.

Rand multiqc

Below is a checklist to step through when evaluating reports.

1. Are the fields correctly populated within the header section of the report
2. Each sample has a prefilter minimum of 30M reads for stranded mRNA, 50M for total RNA, and 25M for low input RNA samples
3. Check duplication rates for samples with higher then average duplication. This is indicative of low complexity libraries. This can be a factor of library construction or the nature of the amplification. It's not unexpected to see lower complexity in low input libraries. Aberrant samples should be brought to the attention of the lab
4. The combination of uniquely mapped and multimapped reads should consist of a higher percent of all reads. Poor mapping is indicative of A) alignment to the incorrect reference or B) sample contamination. Alignment to an incorrect reference is a common occurance while getting acquainted with the workflow.