

RNAseq on the JCU Zodiac HPC

Martha Cooper

2020/06/09. Last updated on 2020-10-01

- [About](#)
- [RNASeq mapping & alignments on the JCU HPC](#)
 - [Step 1: QC of raw fastq files using FASTQC](#)
 - [Step 2: Obtaining a reference genome and annotations](#)
 - [Step 3: Indexing the genome](#)
 - [Step 4: Read Trimming](#)
 - [Step 4: Aligning trimmed reads to the genome](#)
 - [Step 5: Alignment QC](#)
 - [Step 5: Counting Features](#)

About

A collection of information, links and resources for (human) RNA-seq analysis on the Zodiac HPC by researchers in AITHM.

Disclaimer: This documentation is not endorsed by the JCU HPC team/eResearch. I created this resource when I was learning and am sharing it the AITHM researchers in Cairns to (hopefully) make the HPC learning curve a little less steep. Any mistakes are my own.

RNASeq mapping & alignments on the JCU HPC

Notes:

- Some programs use multiple threads or CPUs and where possible this has been utilised. For programs that do not have this functionality, [GNU parallel](#) has been used to parallelise processing (O. Tange (2011): GNU Parallel - The Command-Line Power Tool ;login: The USENIX Magazine, February 2011:42-47).
- PBS script compute resources have been specified for ~35 paired end samples with a read depth of between 50-100 million reads. This translates to 70 raw fastq files.

Compute resource parameters specified in each PBS script may need modification depending on how many files you need to process.

- File paths and environment variables defined in PBS scripts will also need to be changed to match your directory tree.
- Please read all software manuals and make sure that all parameters match specifications for your experiment(s) (try *-help* for more information). This resource has been designed to help you hit the ground running and is very unlikely to represent the perfect use case for your specific experiment.
- Thank you to Ashley Waardenberg who spent a lot of time getting all these softwares up and running on the HPC. This pipeline is based on his work.

Step 1: QC of raw fastq files using FASTQC

FastQC is a popular software that generates QC statistics for next gen seq data. It was written by Simon Andrews from Babraham Bioinformatics. This [great tutorial](#) from Michigan State University explains how to interpret the html output.

FastQC gives 2 outputs for each fastq file; one html document and one zip folder. You can send the html back to your laptop to view the output in a web browser, or **MultiQC** by Phil Ewels can be used to summarise the output of FastQC for all fastq files inputted (Reference: Philip Ewels, Måns Magnusson, Sverker Lundin and Max Källér (2016). MultiQC: summarize analysis results for multiple tools and samples in a single report. Bioinformatics; DOI: 10.1093/bioinformatics/btw354). In this case, we use multiQC to compile all FastQC reports.

```
#!/bin/bash
#PBS -N tRNAFastqc
#PBS -l select=1:ncpus=16:mem=10gb
#PBS -l walltime=2:30:00
#PBS -j oe
#PBS -m abe
#PBS -M martha.cooper@jcu.edu.au

##### Load modules #####
module load fastqc/0.11.7

##### Change to current working directory #####
cd $PBS_O_WORKDIR

##### Set environment vars #####
INPUTDIR=" ../data/RNAseq_27_02_2019/_RAW_DATA/AGRF_CAGRF18155_HHWK3DSXX"
NCPU=15
OUTDIR="$PBS_O_WORKDIR/outputs"
```

```
##### Make output dirs #####
mkdir -p $OUTDIR

##### Execute Program #####
fastqc -t $NCPU -o $OUTDIR $INPUTDIR/*.fastq.gz

##### Load modules for multiQC #####
module load anaconda3
conda init bash
. ~/.bashrc
conda activate multiqc

##### Change to current working directory #####
cd $PBS_O_WORKDIR

##### Execute Program #####
multiqc .
```

- What is `.bashrc`?

Step 2: Obtaining a reference genome and annotations

You can download these straight onto your HPC home directory using `wget` and the following ftp addresses. n.b. more recent genomes and annotations may be available. Make yourself a new directory in called "genome_and_annot", cd into that directory.

Genome (.fa)

- Which genome to use?

The following is from the [STAR manual](#)

"It is strongly recommended to include major chromosomes (e.g., for human chr1-22,chrX,chrY,chrM,) as well as un-placed and un-localized scaffolds. Typically, un-placed/un-localized scaffolds add just a few MegaBases to the genome length, however, a substantial number of reads may map to ribosomal RNA (rRNA) repeats on these scaffolds. These reads would be reported as unmapped if the scaffolds are not included in the genome, or, even worse, may be aligned to wrong loci on the chromosomes. Generally, patches and alternative haplotypes should not be included in the genome. Examples of acceptable genome sequence files: + ENSEMBL: files marked with .dna.primary.assembly, such as: ftp://ftp.ensembl.org/pub/release-77/fasta/homo_sapiens/dna/Homo_sapiens.GRCh38.dna.primary_assembly.fa.gz + GENCODE: files marked with PRI (primary). Strongly recommended for mouse and human: <http://www.gencodegenes.org/>."

At the time of writing, the latest primary assembly for the human genome from Gencode was version 34 and can be obtained here:

```
wget ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_34/GRC
```

Annotations (.gtf)

- Which annotations to use?

The following is from the [STAR manual](#)

“The use of the most comprehensive annotations for a given species is strongly recommended. Very importantly, chromosome names in the annotations GTF file have to match chromosome names in the FASTA genome sequence files. For example, one can use ENSEMBL FASTA files with ENSEMBL GTF files, and UCSC FASTA files with UCSC FASTA files. However, since UCSC uses chr1, chr2, ... naming convention, and ENSEMBL uses 1, 2, ... naming, the ENSEMBL and UCSC FASTA 5 and GTF files cannot be mixed together, unless chromosomes are renamed to match between the FASTA and GTF files. For mouse and human, the Gencode annotations are recommended:

<http://www.gencodegenes.org/>.”

At the time of writing, the latest annotations for the primary assembly from Gencode was version 34 and can be obtained here:

```
wget ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_34/gen
```

Step 3: Indexing the genome

Now we can index the genome using [STAR](#) by Alex Dobin (Ref: Alexander Dobin, Carrie A. Davis, Felix Schlesinger, Jorg Drenkow, Chris Zaleski, Sonali Jha, Philippe Batut, Mark Chaisson, and Thomas R. Gingeras (2013). STAR: ultrafast universal RNA-seq aligner. Bioinformatics; DOI: 10.1093/bioinformatics/bts635). Please read the [manual](#) to make sure the parameters are set correctly for your use. You need to index the genome separately each time you are analysing RNA-seq data with a different read length, where the `--sjdbOverhang` should be set to the read length - 1 (i.e. 150 - 1 = 149)

```
#!/bin/bash
#PBS -N index_hu_150
#PBS -l select=1:ncpus=16:mem=60gb
#PBS -l walltime=2:00:00
#PBS -j oe
#PBS -m abe
#PBS -M martha.cooper@jcu.edu.au

##### Load modules #####
module load star star/2.7.0e

##### Change to current working directory #####
```

```

cd $PBS_O_WORKDIR

##### Unzip genome and annotations #####
gunzip GRCh38.primary_assembly.genome.fa.gz
gunzip gencode.v34.primary_assembly.annotation.gtf

##### Set environment vars #####
THREADS=15
OUTDIR="$PBS_O_WORKDIR/STAR_indexed_150"
GENOME="$PBS_O_WORKDIR/GRCh38.primary_assembly.genome.fa"
ANNOT="$PBS_O_WORKDIR/gencode.v34.primary_assembly.annotation.gtf"

##### Make output dirs #####
mkdir -p $OUTDIR

##### Execute Program #####
STAR --runThreadN $THREADS --runMode genomeGenerate --genomeDir $OUTDIR --g

#### Re-zip the genome #####
gzip GRCh38.primary_assembly.genome.fa

```

Step 4: Read Trimming

The next step is quality and adaptor trimming of the sequencing. [Trim Galore!](#) is a wrapper around Cutadapt and FastQC used for this, written by Felix Krueger.

```

#!/bin/bash
#PBS -N tRNAtrimming
#PBS -l select=1:ncpus=16:mem=10gb
#PBS -l walltime=24:00:00
#PBS -j oe
#PBS -m abe
#PBS -M martha.cooper@jcu.edu.au

##### Load modules #####

module load parallel
module load anaconda3
conda init bash
. ~/.bashrc
conda activate trimgalore

##### Change to current working directory #####
cd $PBS_O_WORKDIR

##### Set environment vars #####
OUTDIR="$PBS_O_WORKDIR/trimmed"

```

```
##### Make output dirs #####
```

```
mkdir -p $OUTDIR
```

```
parallel -j15 --xapply trim_galore --illumina --paired -o $OUTDIR ::: *R1.f
```

- Run FastqQC again to check quality of trimmed reads

```
#!/bin/bash
```

```
#PBS -N tRNAFastqc
```

```
#PBS -l select=1:ncpus=16:mem=10gb
```

```
#PBS -l walltime=3:30:00
```

```
#PBS -j oe
```

```
#PBS -m abe
```

```
#PBS -M martha.cooper@jcu.edu.au
```

```
##### Load modules #####
```

```
module load fastqc/0.11.7
```

```
##### Change to current working directory #####
```

```
cd $PBS_O_WORKDIR
```

```
##### Set environment vars #####
```

```
INPUTDIR="$PBS_O_WORKDIR"
```

```
NCPU=15
```

```
OUTDIR="$PBS_O_WORKDIR/trimmed_fastqc"
```

```
##### Make output dirs #####
```

```
mkdir -p $OUTDIR
```

```
##### Execute Program #####
```

```
fastqc -t $NCPU -o $OUTDIR $INPUTDIR/*.fq.gz
```

```
##### Load modules #####
```

```
module load anaconda3
```

```
conda init bash
```

```
. ~/.bashrc
```

```
conda activate multiqc
```

```
##### Execute Program #####
```

```
multiqc --force . trimmed_fastqc
```

Step 4: Aligning trimmed reads to the genome

STAR is used to align trimmed reads to the indexed reference genome.

- Aligning tRNA reads to complete GTF file

Several non-default options for STAR have been selected in this step. Some represent ENCODE options (see the [STAR manual](#) for more info) and others ensure compatibility for transcriptome assembly using Cufflinks. Read the STAR manual to ensure that you select the correct parameters for your experiment.

```
##### STAR parameters explained #####

### Read mode and file inputs/outputs #####
--runMode alignReads ##mode to run in to align reads
--runThreadN 15 ## no. of threds to use
--sjdbGTFfile $ANNOT ## GTF file with splice junctions in
--genomeDir $INDEXEDGENOMEDIR ## directory of star genome index
--readFilesIn $i ${i%_R1_val_1.fq.gz}_R2_val_2.fq.gz ## read pairs
--readFilesCommand zcat ## command to unzip fa.gz files when reading in
--outFileNamePrefix ${i%_R1_val_1.fq.gz} ## what to name the output files
--outSAMtype BAM Unsorted SortedByCoordinate ## files to output; one unsorted

##### compatibility for cufflinks #####
--outSAMattrIHstart 0 \ # Ensures compatibility with cufflinks downstream
--outFilterIntronMotifs RemoveNoncanonical \ ## Removes non canonical splice

##### encode3 parameters #####
--outFilterMultimapNmax 20
--outFilterMismatchNmax 999
--outFilterMismatchNoverReadLmax 0.04
--outFilterType BySJout
--alignIntronMin 20
--alignIntronMax 1000000
--alignMatesGapMax 1000000
--alignSJoverhangMin 8
--alignSJDBoverhangMin 1
--sjdbScore 1
```

```
#!/bin/bash
#PBS -N alignRNA
#PBS -l select=1:ncpus=16:mem=60gb
#PBS -l walltime=24:00:00
#PBS -j oe
#PBS -m abe
#PBS -M martha.cooper@jcu.edu.au

##### Load modules #####
module load star/2.7.0e

##### Set environment vars #####
INDEXEDGENOMEDIR="/home/jc351340/CHMI_RNAseq/genome_and_annot/STAR_indexed_
```

```

ANNOT="../genome_and_annot/gencode.v34.primary_assembly.annotation.gtf"

##### Change to current working directory #####
cd $PBS_O_WORKDIR

mkdir aligned_tRNA

##### Execute Program #####
for i in *_R1_val_1.fq.gz
do
    STAR --runMode alignReads \
        --runThreadN 15 \
        --sjdbGTFfile $ANNOT \
        --genomeDir $INDEXEDGENOMEDIR \
        --readFilesIn $i ${i%_R1_val_1.fq.gz}_R2_val_2.fq.gz \
        --readFilesCommand zcat \
        --outFileNamePrefix ./aligned_tRNA/${i%_R1_val_1.fq.gz} \
        --outSAMtype BAM Unsorted SortedByCoordinate \
        --outSAMattrIHstart 0 \
        --outFilterIntronMotifs RemoveNoncanonical \
        --outFilterMultimapNmax 20 \
        --outFilterMismatchNmax 999 \
        --outFilterMismatchNoverReadLmax 0.04 \
        --outFilterType BySJout \
        --alignIntronMin 20 \
        --alignIntronMax 1000000 \
        --alignMatesGapMax 1000000 \
        --alignSJoverhangMin 8 \
        --alignSJDBoverhangMin 1 \
        --sjdbScore 1
done

```

This job produces several output files. The info below is from the STAR manual (https://physiology.med.cornell.edu/faculty/skrabanek/lab/angsd/lecture_notes/STARmanual.pdf)

- Aligned.out.bam file - an unsorted bam file. "The paired ends of an alignment are always adjacent, and multiple alignments of a read are adjacent as well. This "unsorted" file can be directly used with downstream software such as HTseq, without the need of name sorting. The order of the reads will match that of the input FASTQ(A) files only if one thread is used -runThread 1, and -outFilterType -BySJout is not used." (STAR manual)
- Aligned.sortedByCoord.out.bam file - bam sorted by coordinate "similar to samtools sort command."
- Log.out - "main log file with a lot of detailed information about the run. This file is most useful for troubleshooting and debugging."

- Log.progress.out - “reports job progress statistics, such as the number of processed reads, %of mapped reads etc. It is updated in 1 minute intervals”
- Log.final.out - “summary mapping statistics after mapping job is complete, very useful for quality control. The statistics are calculated for each read (single- or paired-end) and then summed or averaged over all reads. Note that STAR counts a paired-end read as one read, (unlike the samtools flagstat/idxstats, which count each mate separately). Most of the information is collected about the UNIQUE mappers (unlike samtools flagstat/idxstats which does not separate unique or multi-mappers). Each splicing is counted in the numbers of splices, which would correspond to summing the counts in SJ.out.tab. The mismatch/indel error rates are calculated on a per base basis, i.e. as total number of mismatches/indels in all unique mappers divided by the total number of mapped bases.”
- SJ.out.tab - “contains high confidence collapsed splice junctions in tab-delimited format. Note that STAR defines the junction start/end as intronic bases, while many other software define them as exonic bases.”

The STARtmp folder is supposed to be deleted, but isn't for me (let me know if it is for you!). I did some research and STAR's author Alex says this happens on some systems and [is okay](#) so long as:

- Log.final.out file is generated and is not empty.
- Last line of the Log.out file is “ALL DONE!”

Step 5: Alignment QC

- Alignment QC

[Qualimap 2](#) facilitates QC of alignments in BAM files. (Reference: Konstantin Okonechnikov, Ana Conesa, and Fernando García-Alcalde¹ (2016). Qualimap 2: advanced multi-sample quality control for high-throughput sequencing data. Bioinformatics 32(2): 292–294. DOI: 10.1093/bioinformatics/btv566)

```
#!/bin/bash
#PBS -N qualimapRNA
#PBS -l select=1:ncpus=2:mem=10gb
#PBS -l walltime=6:00:00
#PBS -j oe
#PBS -m abe
#PBS -M martha.cooper@jcu.edu.au

##### Load modules #####
module load parallel
module load anaconda3
conda init bash
```

```

. ~/.bashrc
conda activate qualimap

##### Set environment variables #####
ANNOT="/home/jc351340/CHMI_RNAseq/genome_and_annot/gencode.v34.primary_asse
OUTDIR="$PBS_O_WORKDIR/qualimap_results"

cd $PBS_O_WORKDIR

##### Make output dirs #####
mkdir -p $OUTDIR

##### Execute Program #####

for i in *Aligned.out.bam
do
    mkdir $OUTDIR/${i%Aligned.out.bam}
    qualimap rnaseq \
        -outdir $OUTDIR/${i%Aligned.out.bam} \
        -a proportional \
        -bam $i \
        -p strand-specific-reverse \
        -gtf $ANNOT \
        -pe paired \
        --java-mem-size=8G
done

```

Step 5: Counting Features

HTSeq is a package for counting genomic features in aligned sequencing files and was written by Simon Anders and colleagues. The HTSeq-count function takes a list of genomic features in an annotation file (in gff/gtf format) and aligned sequencing reads (in bam or sam format) and counts how many reads map to each feature.

Reference: Simon Anders, Paul Theodor Pyl, Wolfgang Huber HTSeq — A Python framework to work with high-throughput sequencing data Bioinformatics (2014), in print, online at [doi:10.1093/bioinformatics/btu638](https://doi.org/10.1093/bioinformatics/btu638)

- First, need to sort bam files by name with samtools sort -n (necessary for input to HTSeq-count)

```

#!/bin/bash
#PBS -N bamstRNA
#PBS -l select=1:ncpus=16:mem=50gb
#PBS -l walltime=6:00:00
#PBS -j oe
#PBS -m abe

```

```
#PBS -M martha.cooper@jcu.edu.au

##### Load modules #####
module load samtools
module load parallel

##### Change to current working directory #####
cd $PBS_O_WORKDIR

##### Execute Program #####

## sort by name ##

ls *Aligned.out.bam | parallel -j15 'samtools sort -n {} -o {}.sortedByNa
```

- HTSeq-count. You need to be aware of how your data is stranded. For Illumina Tru Seq data this is stranded on the reverse strand.

tRNA

```
#!/bin/bash
#PBS -N htseq_count
#PBS -l select=1:ncpus=16:mem=10gb
#PBS -l walltime=10:00:00
#PBS -j oe
#PBS -m abe
#PBS -M martha.cooper@jcu.edu.au

##### Load modules #####
module load python3
module load parallel

##### Change to current working directory #####
cd $PBS_O_WORKDIR

##### Execute Program #####
ls *sortedByName.bam | parallel -j15 'python3 -m HTSeq.scripts.count --form
```

And there are you are; counts that can be used for downstream DGE.

Can use multiQC to collate all QC reports:

```
##### Load modules - multiQC #####
##### multiQC will collate report for trimming, aligning and counting #####
module load anaconda3
conda init bash
. ~/.bashrc
```

```
conda activate multiqc
```

```
##### Execute Program #####
```

```
multiqc --force .
```