TOPMed RNA-seq pipeline harmonization summary

This document is intended to ensure reproducibility of analyses and facilitate harmonization of pipelines, defines primary file locations and accessible download locations, and formalizes the TOPMed RNA-seq pipeline components and parameters. All relevant compute parameters are assumed to be software defaults for the versions listed unless specified otherwise (with the exception of directory/file paths, number of threads, etc.).

All wrapper scripts are available from the GTEx pipeline repository: https://github.com/broadinstitute/gtex-pipeline

The current scripts and settings used for TOPMed RNA-seq match commit 1731ed9, packaged here.

Changes to this specification may be proposed based on rigorous benchmarking, pending approval of NHLBI.

Previous versions

The scripts and settings used for the TOPMed MESA RNA-seq pilot match commit 725a2bc, packaged here.

Pipeline summary

The TOPMed RNA-Seq pipeline generates, for each sample:

1. Aligned RNA-seq reads in BAM format.
2. Standard quality control metrics derived from the aligned reads.
3. Gene-level expression quantifications based on a collapsed version of a reference transcript annotation, provided as read counts and TPM.
4. Transcript-level expression quantifications, provided as TPM, expected read counts, and isoform percentages.

This document also describes the generation of the reference files required for each pipeline component.

Pipeline components

- Alignment: STAR 2.6.1d
  - Post-processing: Picard 2.18.17 MarkDuplicates
- Gene quantification and quality control: RNA-SeqC 2.3.3
- Transcript quantification: RSEM 1.3.1
- Utilities: SAMtools 1.9 and HTSlib 1.9

Reference files

This section describes the GRCh38 reference genome and GENCODE 30 annotation used, including the addition of ERCC spike-in annotations.

The reference files described in this section can be obtained through the following links:

- Reference genome for RNA-seq alignment (contains .fasta, .fai, and .dict files): Homo_sapiens_assembly38_noALT_noHLA_noDecoy_ERCC.tar.gz
- Collapsed gene model: gencode.v30.GRCh38.ERCC.genes.collapsed_only.gtf.gz
- STAR index: STAR_genome_GRCh38_noALT_noHLA_noDecoy_ERCC_v30_oh100.tar.gz
RSEM reference: rsem_reference_GRCh38.gencode30.ercc.tar.gz

Note: the reference genome is based on the Broad Institute’s GRCh38 reference, which is used for aligning TOPMed whole genome sequence data.

Genome reference

For RNA-seq analyses, a reference FASTA excluding ALT, HLA, and Decoy contigs was generated (as of the writing of this document, none of the RNA-seq pipeline components were designed to properly handle these regions).

Note: The reference produced after filtering out ALT, HLA, and Decoy contigs is identical to the one used by ENCODE (FASTA file). However, the ENCODE reference does not include ERCC spike-in sequences.

1. The Broad Institute's GRCh38 reference can be obtained from the Broad Institute’s FTP server (ftp://gsapubftp-anonymous@ftp.broadinstitute.org/bundle/hg38/) or from Google Cloud.

2. ERCC spike-in reference annotations were downloaded from ThermoFisher (the archive contains two files, ERCC92.fa and ERCC92.gtf) and processed as detailed below. The patched ERCC references are also available here.
   The ERCC FASTA was patched for compatibility with RNA-SeQC/GATK using
   
   sed 's/ERCC--/ERCC_/g' ERCC92.fa > ERCC92.patched.fa

3. ALT, HLA, and Decoy contigs were excluded from the reference genome FASTA using the following Python code:

   ```python
   with open('Homo_sapiens_assembly38.fasta', 'r') as fasta:
       contigs = fasta.read()
       contigs = contigs.split('>')
       contig_ids = [i.split(' ', 1)[0] for i in contigs]

       # exclude ALT, HLA and decoy contigs
       filtered.fasta = '>'.join([c for i,c in zip(contig_ids, contigs)
                                  if not (i[-4:]=='_alt' or i[:3]=='HLA' or i[-6:]=='_decoy'))]

   with open('Homo_sapiens_assembly38_noALT_noHLA_noDecoy.fasta', 'w') as fasta:
       fasta.write(filtered.fasta)
   ```

4. ERCC spike-in sequences were appended:

   ```bash
   cat Homo_sapiens_assembly38_noALT_noHLA_noDecoy.fasta ERCC92.patched.fa
   > Homo_sapiens_assembly38_noALT_noHLA_noDecoy_ERCC.fasta
   ```

5. The FASTA index and dictionary (required for Picard/GATK) were generated:

   ```bash
   samtools faidx Homo_sapiens_assembly38_noALT_noHLA_noDecoy_ERCC.fasta
   java -jar picard.jar
     CreateSequenceDictionary \
     R=Homo_sapiens_assembly38_noALT_noHLA_noDecoy_ERCC.fasta \
     O=Homo_sapiens_assembly38_noALT_noHLA_noDecoy_ERCC.dict
   ```

Reference annotation

The reference annotations were prepared as follows:

1. The GENCODE 30 annotation was downloaded from the GENCODE FTP.

2. For gene-level quantifications, the annotation was collapsed with the script used in the GTEx pipeline:

   ```bash
   python3 collapse_annotation.py \
     --collapse_only gencode.v30_GRCh38.annotation.gtf \
     gencode.v30_GRCh38.genes.collapsed_only.gtf
   ```
3. Gene- and transcript-level attributes were added to the ERCC GTF with the following Python code:

```python
with open('ERCC92.gtf') as exon_gtf, open('ERCC92.genes.patched.gtf', 'w') as gene_gtf:
    for line in exon_gtf:
        f = line.strip().split('"

        f[0] = f[0].replace('-', '_')  # required for RNA-SeQC/GATK (no '-' in contig name)

        attr = f[8]
        if attr[-1] == ':':
            attr = dict([i.split(' ') for i in attr.replace('"', '').split(' '); '])

        # add gene_name, gene_type
        attr['gene_name'] = attr['gene_id']
        attr['gene_type'] = 'ercc_control'
        attr['gene_status'] = 'KNOWN'
        attr['level'] = 2

        for k in ['id', 'type', 'name', 'status']:
            attr['transcript_str'] = attr['gene'] + k

        attr_str = [k for k in ['gene_id', 'transcript_id', 'gene_type', 'gene_status', 'gene_name',
                                'transcript_type', 'transcript_status', 'transcript_name']
                        if attr[k]
                        .format(k, attr[k])]

        attr_str.append('0:1s"{1:s}".format(k, attr[k])

        f[8] = '".join(attr_str)

    gene_gtf.write('<t'.join(f[:2]+'gene']f[3:]+'\n')
    gene_gtf.write('<t'.join(f[:2]+'transcript']f[3:]+'\n')
    f[8] = '".join(attr_str[:2])
    gene_gtf.write('<t'.join(f[:2]+'exon']f[3:]+'\n')
```

4. The ERCC annotation was appended to the reference GTFs:

```bash
cat gencode.v30 GRCh38.annotation.gtf ERCC92.genes.patched.gtf \
> gencode.v30 GRCh38.annotation.ERCC.gtf
cat gencode.v30 GRCh38.genes.gtf ERCC92.genes.patched.gtf \
> gencode.v30 GRCh38.ERCC.genes.gtf
```

**STAR index**

All RNA-seq samples were sequenced as 2x101bp paired-end, and the STAR index was generated accordingly:

```bash
STAR --runMode genomeGenerate \
    --genomeDir STAR_genome_GRCh38_noALT_noHLA_noDecoy_ERIC_v30_oh100 \
    --genomeFastaFiles Homo_sapiens_assembly38_noALT_noHLA_noDecoy.fasta ERCC92.patched.fa \
    --sjdbGTFfile gencode.v30 GRCh38.annotation.ERCC92.gtf \
    --sjdbOverhang 100 --runThreadN 10
```

**RSEM reference**

The RSEM references were generated using:

```bash
rsem-prepare-reference --num-threads 10 \ 
    --gtf gencode.v30 GRCh38.annotation.ERCC92.gtf \ 
    Homo_sapiens_assembly38_noALT_noHLA_noDecoy fasta, ERCC92.patched.fa \ 
    rsem_reference
```

**Installation of pipeline components**

This section lists the source repositories and installation instructions for the pipeline components. The instruction replicate those in the pipeline **Dockerfile**.

1. STAR v2.6.1d:

```bash
    cd /opt & & 
```
wget --no-check-certificate https://github.com/broadinstitute/picard/releases/download/2.18.17/picard.jar

```bash
2. Picard v2.18.17 or later (for MarkDuplicates):

    mkdir /opt/picard-tools &&
    wget --no-check-certificate
    -P /opt/picard-tools/
    https://github.com/broadinstitute/picard/releases/download/2.18.17/picard.jar
```

```bash
3. RSEM v1.3.1:

    cd /opt &&
    wget --no-check-certificate
    https://github.com/deweylab/RSEM/archive/v1.3.1.tar.gz &&
    tar -xf v1.3.1.tar.gz &&
    rm v1.3.1.tar.gz &&
    cd RSEM-1.3.1 &&
    make
```

```bash
4. RNA-SeQC v2.3.3:

    cd /opt &&
    git clone --recursive https://github.com/broadinstitute/rnaseqc.git &&
    cd rnaseqc &&
    make &&
    make clean
```

**Pipeline parameters**

This section contains the full list of inputs and parameters provided to each method.

The following variables must be defined:

- `star_index`: path to the directory containing the STAR index
- `fastq1` and `fastq2`: paths to the two FASTQ files
- `sample_id`: sample identifier; this will be prepended to output files
- `rsem_reference`: path to the directory containing the RSEM reference
- `genome_fasta`: path to the reference genome
  (Homo_sapiens_assembly38_noALT_noHLA_noDecoy_ERCC.fasta as described above)
- `genes_gtf`: path to the collapsed, gene-level GTF (gencode.v30.GRCh38.ERCC.genes.gtf as described above)
- `star_bam_file`: name/file path of the BAM generated by the STAR aligner, by default
  `{sample_id}.Aligned.sortedByCoord.out.bam`
- `md_bam_file`: name of the BAM generated by Picard MarkDuplicates.

1. **STAR**

    ```bash
    STAR --runMode alignReads \
    --runThreadN 8 \n    --genomeDir $(star_index) \
    --twoPassMode Basic \n    --outFilterMultimapNmax 20 \n    --alignSJoverhangMin 8 \n    --alignSJDNAoverhangMin 1 \n    --outFilterMismatchNmax 999 \n    --outFilterMismatchNoverLmax 0.1 \n    --alignIntronMin 20 \n    --alignIntronMax 1000000 \n    --alignMaxMatesGap 1000000 \n    --outFilterType BySlolot \n    --outFilterScoreMinOverLread 0.33 \n    --outFilterMatchNminOverLread 0.33 \n    --limitSdbInsertNsj 1200000 \n    --readFilesIn \$(fastq1) \$(fastq2) \n    ```
2. MarkDuplicates

```java
java -jar picard.jar 
    MarkDuplicates I=${star_bam_file} 
    O=${md_bam_file} 
    M=${sample_id}.marked_dup_metrics.txt 
    ASSUME_SORT_ORDER=coordinate
```

3. RSEM

```bash
rsem-calculate-expression 
    --num-threads 2 
    --fragment-length-max 1000 
    --no-bam-output 
    --paired-end 
    --estimate-rspd 
    --forward-prob 0.0 
    --bam ${sample_id}.Aligned.toTranscriptome.out.bam 
    ${rsem_reference} ${sample_id}
```

4. RNA-SeQC

```bash
rnaseqc ${genes_gtf} ${bam_file} . 
    --s ${sample_id} --stranded rf --vv
```

**Appendix: wrapper scripts from the GTEx pipeline**

This section provides the commands used to run each step of the pipeline based on the wrapper scripts from the GTEx pipeline.

The following variables must be defined:

- **star_index**: path to the directory containing the STAR index
- **fastq1** and **fastq2**: paths to the two FASTQ files
- **sample_id**: sample identifier; this will be prepended to output files
- **rsem_reference**: path to the directory containing the RSEM reference
- **genome_fasta**: path to the reference genome (Homo_sapiens_assembly38_noALT_noHLA_noDecoy_ERCC.fasta as described above)
- **genes_gtf**: path to the collapsed, gene-level GTF (gencode.v38.GRCh38.ERCC.genes.gtf as described above)

1. STAR (**run_STAR.py**)

```bash
python3 run_STAR.py 
    ${star_index} ${fastq1} ${fastq2} ${sample_id} 
    --output_dir star_out 
    --outputFilterMultimapNmax 20 
    --alignSJoverhangMin 8 
    --alignSJdbOverhangMin 1 
    --outputFilterMismatchNmax 999 
    --outputFilterMismatchNoverLmax 0.1 
    --alignIntronMin 20 
```
2. MarkDuplicates (run_MarkDuplicates.py)

```python
python3 -u run_MarkDuplicates.py ${sample_id}.Aligned.sortedByCoord.out.bam ${sample_id}
```

3. RSEM (run_RSEM.py)

```python
python3 run_RSEM.py \
--max_frag_len 1000 \
--estimate_rspd true \
--is_stranded true \
--threads 2 \
${rsem_reference} ${sample_id}.Aligned.toTranscriptome.out.bam ${sample_id}
```

4. RNA-SeQC (run_rnaseqc.py)

```python
python3 run_rnaseqc.py \
${genes_gtf} \
${sample_id}.Aligned.sortedByCoord.out.md.bam \
${sample_id} \
--stranded rf
```

Certified for TOPMed Production by George J. Papanicolaou, PhD
Data Generation Group Leader and Contracting Officer’s
Representative for NHLBI's Centralized Omics Resource (CORE)
Agreed to By the Broad Institute and the University of
Washington
Issued June 3, 2019