ENCODE Long Read RNA-Seq Analysis Protocol for Human Samples (v3.0)

Prepared by Dana Wyman
March 4, 2020
Ali Mortazavi Lab, University of California, Irvine

Contact Information

Dana Wyman
2300 Biological Sciences III
University of California Irvine
Irvine, CA 92697-2300
Telephone: (949) 824-8393
Email: dwyman@uci.edu

Ali Mortazavi
2218 Biological Sciences III
University of California Irvine
Irvine, CA 92697-2300
Telephone: (949) 824-6762
Email: ali.mortazavi@uci.edu

I. Overview

The long-read RNA-seq data on the ENCODE portal was processed with the ENCODE DCC deployment of the TALON pipeline (available with documentation here: https://github.com/ENCODE-DCC/long-read-rna-pipeline).

This document describes 1) the steps used to generate the fastqs submitted to the DCC, and 2) individual tasks that make up the TALON pipeline as we run them in our lab. The software versions used can be found in Table 1.

Prior to DCC submission, data processing is performed with Circular Consensus (CCS), Lima, and Refine, which are part of the SMRTanalysis software suite available from Pacific Biosciences. They perform the following tasks, respectively:

1. Arrive at a consensus read of insert (ROI) sequence for each cDNA transcript
2. Remove primer/adaptor sequences from the reads
3. Separate full-length ROIs from non full-length. Full-length ROIs are defined by the presence of a polyA tail and an adaptor sequence on each end. This step is very important because it also orients each full-length read to the correct strand using the poly(A) tail as a guide.

The output from Refine consists of full-length, non-chimeric reads in the fastq/fasta format, and these reads form the entry point to the DCC pipeline implementation. From here, the reads are mapped to the genome using Minimap2 to generate output in the sam/bam format. TranscriptClean is run on the mapped reads to correct remaining errors such as noncanonical splice junctions and microindels. Finally, TALON is run to annotate the transcripts and quantify their abundance.

Table 1: Referenced Software

Name	**Version**	**Available from**
DCC TALON pipeline implementation		https://github.com/ENCODE-DCC/long-read-rna-pipeline
CCS	4.0.0	https://github.com/PacificBiosciences/pbbioconda
Lima	1.10.0	https://github.com/PacificBiosciences/pbbioconda
Isoseq3 Refine	3.2.2	https://github.com/PacificBiosciences/pbbioconda
Minimap2	2.17	https://github.com/lh3/minimap2
TranscriptClean	2.0.2	https://github.com/dewyman/TranscriptClean
TALON	5.0	https://github.com/dewyman/TALON
Samtools	1.3	https://sourceforge.net/projects/samtools/files/samtools/1.3/

II. Computational analysis

The programs used in this section are all part of the PacBio SMRTanalysis software suite.

A. Obtaining reads of insert with Circular Consensus (CCS)

Multiple SMRT cells may be sequenced per library in order to get > 1,000,000 raw reads, resulting in a **subreads.bam** file for each SMRT cell. CCS must be run on each of these files individually to generate consensus reads of insert (ROIs).
CCS is run using the following parameters:

```
ccs \
        --noPolish \
        --minLength=10 \
        --minPasses=3 \
        --min-rq=0.9 \
        --min-snr=2.5 \
        --reportFile ccs_report.txt \
        subreads.bam \
        ccs.bam
```

Each CCS run produces a bam file, **ccs.bam**. The sequences in these files are ‘reads of insert’ (ROIs), which represent the consensus sequence of each read.

B. Isolating full-length, non-chimeric reads with Lima and Refine

The purpose of this step is to identify full-length, non-chimeric (FLNC) reads based on the presence of a poly-A tail at the 3’ end and adaptor sequences on each end, and to orient the reads correctly with respect to strand. Non full-length reads are filtered out at this point. A further role of Refine is to identify and filter out chimeric PacBio reads, which form when each end of a SMRTbell adaptor attaches to a different double-stranded cDNA molecule rather than to the blunt ends of the same one.

Lima and Refine are run on each **ccs.bam** file separately using the following parameters:

```
lima ccs.bam \
        PB_adapters.fasta \
        fl.bam \
        --isoseq \
        --num-threads 12 \
        --min-score 0 \
        --min-end-score 0 \
        --min-signal-increase 10 \
        --min-score-lead 0
```

```
isoseq3 refine fl.primer_5p--primer_3p.bam \
                PB_adapters.fasta  \
                flnc.bam \
                --min-polya-length 20 \
                --require-polya \
                --num-threads 12
```

The **flnc.bam** output file contains the full-length, non-chimeric ROIs. We convert **flnc.bam** to the fastq format using the following command:

```
bam2fastq -o flnc -u  flnc.bam
```

The fastq files for all of the SMRT cells are concatenated together and submitted to the DCC . This file represents the starting input for their long read RNA-seq pipeline. Note: the quality scores in this file are not meaningful.

C. Alignment to the reference genome

We next align the FLNC reads to the GRCh38 XY human reference genome (https://www.encodeproject.org/data-standards/reference-sequences/). Run Minimap2 with the following parameters:

```
minimap2 -t 16 -ax splice:hq -uf --MD \
         GRCh38.fa \
         flnc.fastq \
         > Aligned.out.sam
```

The output file will be called **Aligned.out.sam**:

D. Reference-based error correction

I. Extract reference splice junctions

TranscriptClean (next section) requires a file of reference splice junctions in order to correct noncanonical junctions in the PacBio transcripts. To get this file, we run a TranscriptClean utility on the GENCODE v29 comprehensive gene annotation (reference chromosomes only). This file is available here: https://www.gencodegenes.org/human/release_29.html

```
python ${TranscriptCleanPath}/accessory_scripts/get_SJs_from_gtf.py \
    --f ../gencode.v29.annotation.gtf \
    --g GRCh38.fa \
    --o gencode_v29_SJs.tsv
```

The output file **gencode_v29_SJs.tsv**, contains splice junctions derived from the short reads. For each splice junction, it lists genomic location, strand, intron motif, and two additional placeholder columns (to match formatting to a type of STAR splice junction file).

II. Error correction with TranscriptClean

Although the CCS process catches many of the errors found in PacBio transcripts, longer reads and/or those with fewer passes are still prone to mismatch and microindel errors. If these occur on the boundary of an intron, they may create the mistaken appearance of a novel splice junction. TranscriptClean is a Python program we developed to compare the sequences of mapped isoforms to the reference genome and correct likely errors. It can be downloaded from Github at https://github.com/dewyman/TranscriptClean. A file of reference splice junctions for (described in previous section) serves as a reference for correcting junctions. In addition, download a file of all common human variants from dbSNP Build150 (April 2017 release) in the VCF format to run TranscriptClean in variant-aware mode. This file can be found at: https://www.ncbi.nlm.nih.gov/variation/docs/human_variation_vcf/). To make sure that the chromosome naming convention of this file matches the hg38 reference genome, run the following commands:

```
zcat 00-common_all.vcf.gz | \
   awk '{if($0 !~ /^#/ && $0 !~ /^chr/) print "chr"$0; else print $0}' \
   > tmp_00-common_all.vcf
gzip tmp_00-common_all.vcf
mv tmp_00-common_all.vcf.gz 00-common_all.vcf.gz
```

When using variant-aware mode, mismatches and indels are compared to the VCF variant set, and are not modified in the event of a perfect variant match. Unmapped isoforms are discarded by TranscriptClean, and in the case of multi-mapping, only the primary mapping is used. Run TranscriptClean version 2.0.2 on the FLNC reads using the parameters below.

```
python TranscriptClean.py \
        --sam Aligned.out.sam \
        --genome GRCh38.fa \
        --spliceJns gencode_v29_SJs.tsv \
        --correctMismatches true \
        --correctIndels true \
        --variants 00-common_all.vcf.gz \
        --maxLenIndel 5 \
        --maxSJOffset 5 \
        --canonOnly \
        --outprefix libraryID
```

This command will generate two output files: **libraryID_clean.sam** and **libraryID_clean.fa**. These contain the reads with corrections made to remove microindels, mismatches, and noncanonical splice junctions as specified by the parameters. The inclusion of the --canonOnly parameter ensures that all of the reads in the output contain only canonical splice junctions or noncanonical splice junctions that are supported by the reference annotation.

E. Annotate and quantify reads

PacBio is prone to artifacts from internal priming since it uses poly-A selection. In order to screen for this in the data, run the following:

```
talon_label_reads --f libraryID_clean.sam \
   --g GRCh38.fa  \
   --t 16 \
   --ar 20 \
   --deleteTmp \
   --o libraryID
```

This will generate an output SAM file, **libraryID_labeled.sam.** Each read in the output has a tag indicating the fraction of As in the 20-bp interval following the end of the alignment.

In order to compare long read platforms side by side and to track isoforms consistently across multiple datasets, we developed a technology-agnostic long read annotation tool called TALON. It is designed to annotate full-length reads as known or novel transcripts and also to report abundance for these transcripts. Corrected reads are passed into the TALON program, which is built around an SQLite database initialized to contain known genes, transcripts, and exon models from the GENCODE v29 GTF transcriptome annotation.

```
talon_initialize_database \
        --f gencode.v29.annotation.gtf \
        --a gencode_v29 \
        --g mm10 \
        --l 0 \
        --idprefix ENCODEH \
        --5p 500 \
        --3p 300 \
        --o talon.db
```

In a TALON run, each input SAM transcript is compared to the existing transcript models in the database on the basis of its splice junctions, start, and end points. This allows us to not only assign a novel gene or transcript identity where appropriate, but to incorporate new transcript models in the TALON database while characterizing how they differ from known transcript models.

In order to take advantage of TALON's transcript filtering utilities, we run biological replicates through the program together. First, we create a comma-delimited config file (**config.csv**) with metadata about the samples. Column 1 is the unique dataset ID, column 2 is the cell type, column 3 is the platform, and column 4 is the input sam file. Here is an example:

```
Rep1,CellLine,PacBio-Sequel2,Rep1/libraryID1_labeled.sam
Rep2,CellLine,PacBio-Sequel2,Rep2/libraryID2_labeled.sam
```

Next, we run TALON:

```
talon --f config.csv \
                        --db talon.db \
                        --build hg38 \
                        --cov 0.9 \
                        --identity 0.8 \
                        --o cellLine
```

The TALON approach to quantification relies on each long read representing an individual transcript molecule sequenced, which allows us to quantify expression by simply counting the number of reads that were assigned to a particular transcript or gene and then converting these values to units of transcripts per million (TPM). To obtain a raw abundance file, run the following command on the TALON database:

```
talon_abundance \
         --db talon.db \
         -a gencode_v29 \
         --build hg38 \
         --o cellLine
```

The resulting file, **cellLine_talon_abundance.tsv**, can be used to compute gene-level expression values. For gene expression, we include all reads assigned to a locus, since even novel transcripts that did not meet the threshold to become a new transcript model are informative for the overall gene expression level. On the transcript level, however, we apply our TALON filters in order to avoid quantifying transcript models with insufficient evidence. Our filtering process uses the novelty labels assigned to each observed transcript model in order to remove likely artifacts. Observed transcripts that fully match counterparts in the GENCODE annotation are accepted immediately, but novel transcripts must be reproducibly detected at least 5 times in each biological replicates in order to be included in the downstream analysis. In addition, reads annotated with a putative internal priming event (> 0.5 As in 20bp after alignment) are excluded from the filtering process. Genomic transcripts are always removed, since they are not likely to constitute real isoforms. **Note: if biological replicates are not available for the sample in question, replicate-based filtering of novel transcripts is not possible. Treat these results with caution.**

To obtain a filtered abundance file for the datasets, run the following commands to first create a transcript whitelist and then apply it during abundance table generation:

```
talon_filter_transcripts \
        --db talon.db \
        -a gencode_v29 \
        --maxFracA 0.5 \
        --minCount 5 \
        --minDatasets 2 \
        --datasets Rep1,Rep2 \
        --o whitelist.csv
```

```
talon_abundance \
        --db talon.db \
        -a gencode_v29 \
        --build hg38 \
        --whitelist whitelist.csv \
        -d Rep1,Rep2 \
        --o cellLine
```

It is also possible to generate a custom GTF transcriptome annotation for the samples from the TALON database. This file contains only transcript models from those samples that passed the TALON filters. The start and end coordinates used for these models are the ones first recorded in the database, so for known transcripts, that means the ones from GENCODE. To generate a filtered GTF, use the transcript whitelist we generated earlier:

```
talon_create_GTF \
        --db talon.db \
        -a gencode_v29 \
        -b hg38 \
        --whitelist whitelist.csv \
        --o cellLine_filtered
```

IV. References

1. Li, H. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics* **34**, 3094–3100 (2018).

2. Gordon, S. P. et al. Widespread Polycistronic Transcripts in Fungi Revealed by Single-Molecule mRNA Sequencing. *PLoS ONE* 10, e0132628 (2015).

3. Li H.*, Handsaker B.*, Wysoker A., Fennell T., Ruan J., Homer N., Marth G., Abecasis G., Durbin R. and 1000 Genome Project Data Processing Subgroup (2009) The Sequence alignment/map (SAM) format and SAMtools. Bioinformatics, 25, 2078-9.

4. Wyman, D., TranscriptClean: A program for correcting mismatches, microindels, and noncanonical splice junctions in long reads, (2018), GitHub repository, https://github.com/dewyman/TranscriptClean

5. Wyman, D., TALON: Technology agnostic long read analysis pipeline for transcriptomes, (2019), GitHub repository, https://github.com/dewyman/TALON