Oct 13, 2020 Version 12

Stranded Transcript Count Table Generation from Long Reads V.12

Version 1 is forked from Transcript Coverage Analysis from Long Reads

DOI
dx.doi.org/10.17504/protocols.io.bndema3e

David A Eccles[1]

[1]Malaghan Institute of Medical Research (NZ)

David A Eccles
Malaghan Institute of Medical Research (NZ)

OPEN ACCESS

DOI: dx.doi.org/10.17504/protocols.io.bndema3e

Protocol Citation: David A Eccles 2020. Stranded Transcript Count Table Generation from Long Reads. **protocols.io https://dx.doi.org/10.17504/protocols.io.bndema3e**Version created by **David A Eccles**

Protocol status: In development
We are still developing and optimizing this protocol

Created: October 13, 2020

Last Modified: October 13, 2020

Protocol Integer ID: 43142

Abstract

This protocol is for comparing different samples at the transcript level, using long reads that are mapped to transcripts.

Input(s): demultiplexed and oriented fastq files (see protocol **Preparing Reads for Stranded Mapping**), transcript reference fasta file, annotation file

Output(s): transcript table, sorted by differential coverage, annotated with gene name / description / location

Before start

Obtain a transcript fasta file, and an annotation file. For the mouse genome, I use the following files:

1. Transcript sequences from **Ensembl**; this should be the union of cDNA, CDS, and ncRNA sequences (e.g. from **This directory**).
2. Annotation file obtained from **Ensembl BioMart** (Ensembl Genes → Mouse Genes) as a compressed TSV file with the following attribute columns:

- Transcript stable ID
- Gene name
- Gene description
- Chromosome/scaffold name
- Gene start (bp)
- Gene end (bp)
- Strand

A recent version of these files can be obtained from **This Zenodo Repository**

Demultiplex Reads

1

Demultiplex and orient reads as per the protocol **Preparing Reads for Stranded Mapping**. It is expected that these demultiplexed reads will be split up in the current directory, and coupled with a '*barcode_counts.txt*' file. If that's the case, the following should work:

```
for bc in $(awk '{print $2}' barcode_counts.txt);
  do ls oriented/${bc}_reads_dirAdjusted.fq.gz;
done
```

Example expected output:

```
oriented/BC03_reads_dirAdjusted.fastq.gz
oriented/BC04_reads_dirAdjusted.fastq.gz
oriented/BC05_reads_dirAdjusted.fastq.gz
oriented/BC06_reads_dirAdjusted.fastq.gz
oriented/BC07_reads_dirAdjusted.fastq.gz
oriented/BC08_reads_dirAdjusted.fastq.gz
```

If the '*barcode_counts.txt*' file is not present, this error will appear:

```
awk: fatal: cannot open file `barcode_counts.txt' for reading (No
such file or directory)
```

If one or more of the oriented read files is missing, it will look something like this:

```
oriented/BC03_reads_dirAdjusted.fastq.gz
oriented/BC04_reads_dirAdjusted.fastq.gz
ls: cannot access 'oriented/BC05_reads_dirAdjusted.fastq.gz':
  No such file or directory
ls: cannot access 'oriented/BC06_reads_dirAdjusted.fastq.gz':
  No such file or directory
oriented/BC07_reads_dirAdjusted.fastq.gz
oriented/BC08_reads_dirAdjusted.fastq.gz
```

Index Preparation

2 Prepare a substitution matrix for barcode mapping. The default substitution matrix is swayed too much by INDELs in the barcode sequences, so here's one that I've developed using a combination of trial & error and last-train:

```
#last -Q 0
#last -a 13
#last -A 13
#last -b 4
#last -B 4
#last -S 1
# score matrix (query letters = columns, reference letters =
rows):
       A      C      G      T
A      5    -18     -7    -18
C    -18      6    -18    -12
G     -7    -18      5    -18
T    -18    -12    -18      6
```

cDNA.mat

*[note: this is a **different** matrix from that used for demultiplexing and read orientation]*

3 Prepare transcript index (see Guidelines for data sources). Following **Martin Frith's recommendation**, the '-uNEAR' seeding scheme is used to slightly increase sensitivity. This will generate seven additional files of the form <index name>.XXX:

```
lastdb -uNEAR Mus_musculus.GRCm38.ensembl_v98.mtr.fa
Mus_musculus.GRCm38.ensembl_v98.mtr.fa
```

Transcriptome Mapping

4 Reads are mapped to the transcriptome with LAST.

The results of that mapping can be piped through *last-split* and *last-postmask* to exclude unlikely hits, then through '*maf-convert -n tab*' to convert to a one-line-per-mapping CSV format. This CSV format is further processed to make sure that there is only one mapping per transcript-read pair.

```
mkdir -p mapped
for bc in $(awk '{print $2}' barcode_counts.txt);
  do echo "** ${bc} **";
  lastal -P 10 -p cDNA.mat Mus_musculus.GRCm38.ensembl_v98.mtr.fa
<(pv oriented/${bc}_reads_dirAdjusted.fq.gz | zcat) | \
    last-split -n -m0.99 | last-postmask | maf-convert -n tab | \
    cut -f 2,7,10 | sort | \
    uniq | gzip >
mapped/trnMapping_LAST_${bc}_vs_Mmus_transcriptome.txt.gz;
done
```

5 The result is then aggregated to sum up counts per transcript:

```
for bc in $(awk '{print $2}' barcode_counts.txt);
  do echo "** ${bc} **";
  zcat mapped/trnMapping_LAST_${bc}_vs_Mmus_transcriptome.txt.gz |
\
    awk -F'\t' -v "bc=${bc}" '{print bc,$1,$3}' | sort | uniq -c |
\
    gzip >
mapped/trnCounts_LAST_${bc}_vs_Mmus_transcriptome.txt.gz;
done
```

Note: I've split this up into two steps (compared to previous versions of this protocol) so that an intermediate count of the total number of mapped transcripts per barcode can be done:

```
for bc in $(awk '{print $2}' barcode_counts.txt);
  do echo -n "${bc} ";
  zcat mapped/trnMapping_LAST_${bc}_vs_Mmus_transcriptome.txt.gz |
\
    awk '{print $2}' | sort | uniq | wc -l;
done
```

Annotation and Result generation

6 count_analysis.r

Transcript counts are merged with ensembl gene annotation, then converted into wide format (one line per transcript) using an R script.

The transcript annotation in this case is from ensembl BioMart (see Guidelines for more details).

```
#!/usr/bin/env Rscript

library(tidyverse);

## load used barcode identifiers
bcNames <- read.table("barcode_counts.txt",
stringsAsFactors=FALSE)[,2];

## load count data into "narrow" array (one line per count)
trn.counts <- tibble();
for(bc in bcNames){
    trn.counts <-
        bind_rows(trn.counts,

sprintf("mapped/trnCounts_LAST_%s_vs_Mmus_transcriptome.txt.gz",
bc) %>%
              read_table2(col_names=c("count","barcode",
                                      "transcript","dir")));
}

## remove revision number from transcript names (if present)
trn.counts$transcript <- sub("\\.[0-
9]+$","",trn.counts$transcript);

## convert to wide format (one line per transcript)
trn.counts.wide <- spread(trn.counts, barcode, count) %>%
    mutate(dir = c("+"="fwd", "-"="rev")[dir]);
for(bd in colnames(trn.counts.wide[,-1])){
    trn.counts.wide[[bd]] <- replace_na(trn.counts.wide[[bd]],0);
}

## load ensemble transcript metadata (including gene name)
ensembl.df <-
read_delim("ensembl_mm10_geneFeatureLocations.txt.gz",
                         delim="\t");

colnames(ensembl.df) <-
    c("Transcript stable ID" = "transcript",
      "Gene description" = "Description",
      "Gene name" = "Gene",
      "Gene start (bp)" = "Start",
      "Gene end (bp)" = "End",
      "Strand" = "Strand",
      "Chromosome/scaffold name" = "Chr")[colnames(ensembl.df)];
```

```
ensembl.df$Description <- sub(" \\
[.*$","",ensembl.df$Description);
ensembl.df$Description <- sub("^(.
{50}).+$","\\1...",ensembl.df$Description);
options(scipen=15); ## don't show scientific notation for large
positions

## merge ensembl metadata with transcript counts
gene.counts.wide <- inner_join(ensembl.df, trn.counts.wide,
by="transcript");
gene.counts.wide <- gene.counts.wide[order(-
rowSums(gene.counts.wide[,-(1:8)])),];
## write result out to a file
write.csv(gene.counts.wide,
file="wide_transcript_counts_LAST.csv", row.names=FALSE);
```

Downstream Workflows

7 Here is a downstream workflow that carries out transcript-level differential expression analysis using **DESeq2**:

- **Creating Differential Transcript Expression Results with DESeq2**

I would like to emphasise that batch effects should be considered for nanopore sequencing, given how frequently the technology changes. Make sure that at least the sequencing *library* (i.e. samples prepared in tandem on the same day from the same kit) is added into the statistical model, and try to make sure that sequencing libraries are fairly heterogeneous - replicates from a sample with skewed transcript distributions could influence the outcome of statistical tests.