Chapter 14

Bioinformatics Pipeline for Transcriptome Sequencing Analysis

Sarah Djebali, Valentin Wucher, Sylvain Foissac, Christophe Hitte, Erwan Corre, and Thomas Derrien

Abstract

The development of High Throughput Sequencing (HTS) for RNA profiling (RNA-seq) has shed light on the diversity of transcriptomes. While RNA-seq is becoming a de facto standard for monitoring the population of expressed transcripts in a given condition at a specific time, processing the huge amount of data it generates requires dedicated bioinformatics programs. Here, we describe a standard bioinformatics protocol using state-of-the-art tools, the STAR mapper to align reads onto a reference genome, Cufflinks to reconstruct the transcriptome, and RSEM to quantify expression levels of genes and transcripts. We present the workflow using human transcriptome sequencing data from two biological replicates of the K562 cell line produced as part of the ENCODE3 project.

Key words Transcriptome sequencing, Protocols, RNA-seq, Bioinformatics workflow

1 Introduction

1.1 RNA-Seq Technology

The application of HTS technologies for cDNAs (RNA-seq) allows to characterize the myriad of RNA molecules transcribed in a given cell or tissue at a specific time point [1]. The so-called transcriptome sequencing provides a unique snapshot of all expressed transcripts in a particular condition and thus informs about fundamental biological processes such as (1) the transcribed coding (mRNAs) and noncoding RNAs (ncRNAs), (2) novel alternative isoforms, chimeric genes/transcripts, and (3) the levels of expression of these RNAs.

Depending on the experimental protocol, RNA-seq can be used to target specific categories of RNAs based for instance on their sizes (long vs. short RNAs), their molecular properties (RNAs

The original version of this chapter was revised. An erratum to this chapter can be found at DOI 10.1007/978-1-4939-4035-6_17

Ulf Andersson Ørom (ed.), *Enhancer RNAs: Methods and Protocols*, Methods in Molecular Biology, vol. 1468,
DOI 10.1007/978-1-4939-4035-6_14, © Springer Science+Business Media New York 2017

with a polyA tail vs. ribosomal RNAs), or their cellular compartments (cytoplasmic vs. nuclear RNAs) [2].

However, most of the active sequencing platforms worldwide currently rely on a technology that generally does not produce the entire sequence of a nucleic acid: the process can only generate sequences—called "reads"—of a limited length from the extremities of each RNA molecule. Therefore a fragmentation step is generally included in the protocol in order to allow any position of the transcript to be potentially sequenced. When both extremities of each fragment are read the process is called Paired-End sequencing, and pairs of reads are obtained. The read length and the size distribution of the sequenced fragments are important features of the process.

Typically, tens of million of reads of 100–200 bp are generally produced from fragments of about 200–400 bp. These reads need to be:

- Mapped back onto a reference genome taking into account splice sites (mapping step).
- Assembled into exon–intron structures (transcriptome reconstructions step).
- Used to quantify known and/or novel transcripts or genes (quantification step).

However, given the depth provided by current sequencing machines and although numerous and efficient bioinformatics tools dedicated to this task exist, dealing with such massive amounts of data remains a challenge.

1.2 The STAR: Cufflinks: RSEM pipeline

Here we present a commonly used bioinformatics pipeline to process RNA-seq reads using STAR [3] for mapping sequences, Cufflinks [4] for transcript model reconstruction and RSEM [5] for transcript and gene quantifications (*see* Fig. 1). This pipeline quantifies annotated genes and transcripts, however the commands we provide are general enough to be easily extended to quantify both known and novel transcripts.

These programs, actively maintained by their developers, are widely used by the community including international consortia such as ENCODE3 [6], Blueprint [7] or TCGA (http://cancergenome.nih.gov). Moreover, benchmarks done as part of the RGASP project (RNA-seq Genome Assessment Project) [8, 9], or as part of the ENCODE3 evaluation (under review), show that they yield favorable performances while limiting computational needs.

1.3 Alternative Pipelines

Nevertheless, there are many alternative programs that could be used at each step of the workflow with for instance tophat2/bowtie2 [10] or the GEMtools RNA-seq pipeline [11] for splice-aware read mapping software, stringtie [12] for transcriptome reconstruction, and Flux capacitor [13], eXpress [14] or Sailfish [15] for transcript and gene quantifications.

In the following tutorial, we assume that both a reference genome sequence and a genome annotation are available, allowing

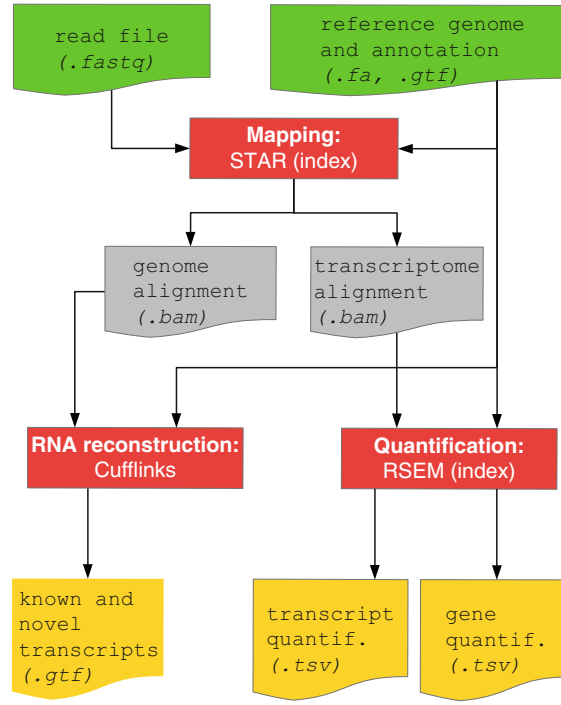

Fig. 1 Pipeline description. Schematic overview of the bioinformatics pipeline described in this protocol. Input files are in *green*, intermediary files in *gray*, output files in *yellow,* and the main steps (bioinformatics tools) in *red*. Using reference genome and annotation files, RNA-seq reads are mapped using STAR to the genome and to the transcriptome. The genome alignment output file is then used by Cufflinks to reconstruct known and novel transcripts. The transcriptome alignment output file is used by RSEM to quantify the levels of expression of genes and transcripts. The index construction required by STAR and RSEM is implicitly represented

the user to implement a genome-guided assembly protocol. In the absence of a reference genome sequence, one would favor de novo transcriptome assembly which involves different algorithms, such as for instance, Trinity [16] or KisSplice [17]. In addition, the biological and bioinformatics protocols may vary with respect to the sequencing technology and the species to be studied. Here, we illustrate the method which uses Illumina paired-end reads (cf. **Note 1**) from vertebrate species samples.

2 Materials

While genome-guided algorithms for transcriptome assembly tend to limit computational resources (which would not be the case for de novo transcriptome assembly), minimal computational resources are still required. We have tested this protocol using a 64-bit Linux system with 32Go of RAM and 8 cores.

In this protocol, all command lines will be written in Courier New police.

Moreover, in order to distinguish biological materials (sequenced reads, reference genome and annotation files) from bioinformatics software, all input "biological files" will be stored in a specific directory named material.

To create this directory, type the following command line:

```
mkdir material
```

2.1 RNA-Seq FASTQ Reads

For this tutorial, we use two biological replicates of the human K562 cell line (adult 53 year female) from the human ENCODE3 RNA evaluation project. The files are available here:

https://www.encodeproject.org/experiments/ENCSR000AEM/

The libraries come from two independent growths of the K562 cell line and the sequencing was done using Illumina Hi-Seq technology with a stranded paired-end read protocol. Only polyA+ RNAs with a size greater than 200 nucleotides were selected, naturally leading to a ribosomal RNA depletion.

Since there are two biological replicates, four read files are available (i.e., two mates x two replicates) with the following IDs ENCFF001RDZ.fastq.gz and ENCFF001RED.fastq.gz for replicate 1 and ENCFF001REF.fastq.gz and ENCFF001REG.fastq.gz for replicate 2. For clarity, we will add a \ when the command line is too long. To download the compressed read files:

```
wget -O ENCFF001RDZ.fastq.gz \
https://www.encodeproject.org/files/
   ENCFF001RDZ/@@download/ENCFF001RDZ.fastq.gz
wget -O ENCFF001RED.fastq.gz \
https://www.encodeproject.org/files/
   ENCFF001RED/@@download/ENCFF001RED.fastq.gz
wget -O ENCFF001REF.fastq.gz \
https://www.encodeproject.org/files/
   ENCFF001REF/@@download/ENCFF001REF.fastq.gz
wget -O ENCFF001REG.fastq.gz \
https://www.encodeproject.org/files/
   ENCFF001REG/@@download/ENCFF001REG.fastq.gz
```

Then, move these four files into the material directory:

```
mv ENCFF001*fastq.gz materials/
```

2.2 Reference Genome and Annotation Files

We will use the human genome assembly version hg19 (aka GRCh37) available as a multifasta file (each sequence corresponding to one chromosome) from the UCSC website [18] here:

```
wget http://hgdownload.cse.ucsc.edu/golden-
   Path/hg19/encodeDCC/referenceSequences/male.
   hg19.fa.gz
```

To decompress the file:

```
gunzip male.hg19.fa.gz
```

Please, note that this file corresponds to the primary assembly of the human male genome, i.e., without haplotypic chromosomes but including chromosome Y (*see* **Note 2**). Repeat sequences are soft-masked which means that all repeats and low complexity regions have been replaced with the lowercase version of their nucleic base.

The reference genome annotation used in this tutorial will be GENCODE [19] which is part of the ENCODE project and whose aim is to annotate all evidence-based gene features on the human genome. The GENCODE gene set is actually the reference human gene annotation used by international projects (ENCODE, 1000 genomes…), its main advantages being its comprehensiveness, since it includes both long noncoding RNA [20] and pseudogene [21] annotations. Providing such a reliable data to the process is fundamental, as several steps of the pipeline (mapping, transcript building, and quantification) are affected by the quality of the annotation.

Generally, the annotation file is stored in a .GFF or GFF3 (General Feature Format) or a .GTF/GFF2.5 (General Transfer Format corresponding to the GFF2.5) which is a 9 columns tab-delimited file storing informations (localization, source, transcriptional orientation) on specific features (gene, transcript, exon, etc.).

Importantly, the version of the annotation must correspond to the genomic sequence used in the pipeline: make sure that the gencode file version matches with the genome assembly (*see* **Note 3**). Here, we retrieved gencode version 19 from (built on the hg19 human genome assembly) usingthe gencode FTP website (ftp://ftp.sanger.ac.uk/pub/gencode/Gencode_human/):

```
wget
ftp://ftp.sanger.ac.uk/pub/gencode/Gencode_hu-
   man/release_
19/gencode.v19.annotation.gtf.gz
gunzip gencode.v19.annotation.gtf.gz
```

Then, both reference files (genome and annotation) are moved to the material directory:

```
mv male.hg19.fa gencode.v19.annotation.gtf ma-
terial/
```

In order to ease the access to these files, it is recommended to create two shell variables ($GENOME and $ANNOTATION for genome and annotation files, respectively) which will point to the absolute path of the files:

```
GENOME=$(readlink -f ./material/male.hg19.fa)
ANNOTATION=$(readlink -f ./material/gencode.
v19.annotation.gtf)
```

2.3 Software Installation

The bioinformatics programs used in this tutorial will be stored in a specific directory named bin. To create this directory and move there, type the following:

```
mkdir bin
cd bin
```

One essential step in each program installation is to make sure that the directory where the program has been installed (or is located) is present in your PATH environment. This dynamic variable lists all the directories that the shell searches through when the user tries to execute a program, thus avoiding the need to use the full path to the program.

2.4 SAMtools

SAMtools [22] are a suite of utilities for manipulating alignments in SAM format (Sequence Alignment/Map) which is the standard format for storing large nucleotide alignments (typically, those encountered in HTS sequencing). In addition, SAMtools are also required by some of the programs described in this chapter such as cufflinks for instance. To install SAMtools, one needs to have zlib and htslib installed.

Check the latest version of the SAMtools on the dedicated website in order to download it:

```
wget -O samtools-1.2.tar.bz2 \ https://github.
   com/samtools/samtools/releases/download/1.2/
   samtools-1.2.tar.bz2
tar xjvf samtools-1.2.tar.bz2
cd samtools-1.2
make
```

As mentioned above, the resulting "samtools" binary is added to the user's PATH environment variable using the following command line where $PWD corresponds to the full path of the current working directory. Please note that this command line assumes a Bourne-Again shell interpreter (bash, *see* **Note 4**).

```
export PATH=$PATH:${PWD}
```

2.5 STAR

Download the latest version of the STAR mapper [3] freely available from its github website: https://github.com/alexdobin/STAR

```
wget -O STAR_2.5.0a.tar.gz \ https://github.
   com/alexdobin/STAR/archive/STAR_2.5.0a.tar.
   gz
tar zxvf STAR_2.5.0a.tar.gz
cd STAR-STAR_2.5.0a/
```

Here, it is either possible to use the precompiled binaries in the ./bin/ directory or to compile the sources such as:

```
cd source
make STAR
export PATH=$PATH:${PWD}
```

2.6 Cufflinks

Like STAR, Cufflinks [4] can either be installed using a precompiled binary release or built from the sources (note that the latter option also requires to install the Boost C++ libraries).

Both Linux and Mac versions are available. Here we download the latest binary version 2.2.1 and then export the cufflinks executable in the PATH.

```
wget http://cole-trapnell-lab.github.io/cuf-
   flinks/assets/downloads/cufflinks-2.2.1.Linux_
   x86_64.tar.gz
tar xzvf cufflinks-2.2.1.Linux_x86_64.tar.gz
export PATH=$PATH:${PWD}
```

2.7 RSEM

For RSEM [5], download the latest archive available on the github website (https://github.com/deweylab/RSEM) and then add it to your PATH. Note that for compatibility with STAR2.5 it is essential to use a version of RSEM that is at least 1.2.25.

```
wget -O RSEM.v1.2.25.tar.gz \ https://github.
   com/deweylab/RSEM/archive/v1.2.25.tar.gz
cd RSEM-1.2.25/
make
export PATH=$PATH:${PWD}
```

3 Methods

Before starting to process the sequence reads, and if this task has not already been performed by the sequencing platform, it is always relevant to assess their quality (*see* **Note 5**). This tutorial demonstrates how to run each step of the pipeline separately, allowing to self- tune each step's parameters with respect to users' needs and to understand problems when they arise. Note that for the mapping and the known gene and transcript quantification parts of the pipeline, bash script and nextflow implementations also exist (*see* **Notes 6** and **7**). In case the RNA-seq experiment has been performed using control RNA spike-ins, they can be used by slightly changing the following protocol (*see* **Note 8**).

3.1 Mapping

STAR uses a suffix array approach to map reads to the genome and to the annotated splice junctions. Reads can be mapped both in a continuous way, i.e., in one block, or in a noncontinuous way, i.e., allowing gaps which can be considered as introns if long enough (see --alignIntronMin option below), in which case the read mapping is called a split-mapping.

STAR

3.1.1 Making the STAR Indices

This only needs to be done once for a given project. The same indices can then be used for all RNA-seq datasets of this project.

3.1.2 Input Files and Arguments

$STARgenomeDir is the directory where the STAR indices will be stored. This directory has to be created with mkdir and given write permissions before the command is run.

- $GENOME is the genome FASTA file.
- $ANNOTATION is the gene annotation in GTF format.
- $threads is the number of threads for parallelizing the task. Here, we fixed it to 8 given the computing resources available, but if one can only use 4 the job will simply take longer (See Mat.).

3.1.3 Command

```
STAR --runThreadN $threads --runMode genomeGen-
erate \
--genomeDir $STARgenomeDir --genomeFastaFiles
   $GENOME \
--sjdbGTFfile $ANNOTATION --sjdbOverhang 100 \
--outFileNamePrefix $STARgenomeDir
```

Note that the --sjdbOverhang option corresponds to the length of the genomic sequence around the annotated junction to be used in constructing the splice junction database, and should be set to the read length minus one.

3.1.4 Output Files

The above command will generate many genome files in the directory $STARgenomeDir, most of which use internal STAR format and are not intended to be utilized by the end user. None of them should be changed. The chrNameLength.txt file contains the chromosome names and lengths and is useful to generate RNA-seq signal files in bigwig format (https://genome.ucsc.edu/goldenpath/help/bigWig.html) from the continuous valued bedgraph files produced by STAR (see below).

3.1.5 Mapping the Reads

Mapping of the reads has to be performed for each RNA-seq dataset of a given project. STAR creates a genome mapping file and a transcriptome mapping file successively, by internally converting genome global coordinates to transcript local coordinates.

3.1.6 Input Files and Arguments

- $STARgenomeDir is the STAR index file directory (see above).
- $read1 is the gzipped FASTQ file of the first mates.
- $read2 is the gzipped FASTQ file of the second mates.
- $nThreadsSTAR is the number of threads.

3.1.7 Command

```
STAR --genomeDir $STARgenomeDir --readFilesIn
$read1 $read2 \
--readFilesCommand zcat --outFilterType
   BySJout --outSAMunmapped Within \
--outSAMtype BAM SortedByCoordinate --outSA-
   MattrIHstart 0 \
--outFilterIntronMotifs RemoveNoncanonical
   --runThreadN $nThreadsSTAR \
--quantMode TranscriptomeSAM --outWigType bed-
   Graph --outWigStrand Stranded
```

The --readFilesCommand zcat option is used to uncompress gzipped read fastq files provided as input, while the --outFilterType BySJout option is used to reduce the number of spurious junctions. The --outSAMunmapped Within and the --outSAMtype BAM SortedByCoordinate are used to produce a standard bam file sorted by coordinates, while the --outSAMattrIHstart 0 and the --outFilterIntronMotifs RemoveNoncanonical are important to produce an output file that is compatible and better to use for cufflinks, respectively. The --quantMode TranscriptomeSAM is used to produce a transcriptome bam file that will be used by RSEM, while the --outWigType bedGraph and the --outWigStrand Stranded optionsproduce stranded continuous valued bedgraph files from the alignments.

STAR allows the user to specify many other options, which can be found in the STAR manual (https://github.com/alexdobin/STAR/blob/master/doc/STARmanual.pdf), however a shorter list of such options, currently used in ENCODE3 and for which we use default values here, are provided in Table 1.

3.1.8 Output Files

STAR produces many output files within the current working directory (note that the output directory can be changed with the --outFileNamePrefix option), the most important of which are the following eight:

- Log.final.out: summary mapping statistics, useful for quality control, with the number and percentage of initial fragments (read pairs) that were mapped, the number and percentage of fragments that are mapped uniquely, that are mapped multiple times (called multimaps, split into the different reasons for that), and unmapped (summarized in Table 2), as well as statistics about splice junction detection.
- Aligned.sortedByCoord.out.bam: the genome BAM file sorted by coordinates.
- Aligned.toTranscriptome.out.bam: the transcriptome BAM file. This file is not sorted, which, in case several threads are used, does not guarantee exact reproducibility of the downstream RSEM quantifications. If exact reproducibility is wanted

Table 1

STAR options which differ between this tutorial and ENCODE3

Option	Meaning	Default value	Encode value
--outSAMattributes	A string of desired SAM attributes, in the order desired for the output SAM	NH HI AS nM	NH HI AS NM MD
--outFilterMultimapNmax	Read alignments will be output only if the read maps fewer times than this value, otherwise no alignments will be output	10	20
--outFilterMismatchNmax	Alignment will be output only if it has fewer mismatches than this value	10	999
--outFilterMismatchNoverReadLmax	Alignment will be output only if its ratio of mismatches to read length is less than this value	1	0.04
--alignIntronMin	Minimum intron size: genomic gap is considered intron if its length>=alignIntronMin, otherwise it is considered Deletion	21	20
--alignIntronMax	Maximum intron size: if 0, max intron size will be determined by (2^winBinNbits)*winAnchorDistNbins	0	1000000
--alignMatesGapMax	Maximum gap between two mates, if 0, max intron gap will be determined by (2^winBinNbits)*winAnchorDistNbins	0	1000000
--alignSJoverhangMin	Minimum overhang (block size) for spliced alignments	5	8
--alignSJDBoverhangMin	Minimum overhang (block size) for annotated spliced alignments	3	1
--sjdbScore	Extra alignment score for alignments that cross database junctions	2	1
--genomeLoad	Mode of shared memory usage for the genome files	NoSharedMemory	LoadAndKeep
--limitBAMsortRAM	Maximum available RAM for sorting BAM. If 0, it will be set to the genome index size. 0 value can only be used with --genomeLoad NoSharedMemory option	0	10000000000

For each such option we provide its name, its meaning, its default value (used here), and the value used in ENCODE3. The mapping results used with one set of values or the other do not vary drastically, and lead to very similar gene and transcript quantifications

Table 2

Mapped fragment statistics

	Mapped		Uniquely mapped		Multi-mapped	
# Fragments	#	%	#	%	#	%
113,327,735	103,116,159	91.0	99,717,493	88.0	3,398,666	3.0

Number of initial, mapped, uniquely mapped, and multi-mapped fragments are provided for the RNA-seq experiment under study

and more than 1 thread is used, this file has to be sorted (as is done in https://github.com/ENCODE-DCC/long-RNASeq-pipeline/blob/master/DAC/STAR_RSEM.sh).

– SJ.out.tab: contains high confidence collapsed splice junctions derived from the split-mapped reads.

– Signal.[type].str[no].out.bg, where type is the type of alignment (either Unique or UniqueMultiple), and where no is the strand number (either 1 or 2), are four stranded bedgraph (BG) files made from the alignments and that can be converted into bigwig files (BW) using the bedGraphToBigWig UCSC tool, for visualization of the mapped read coverage over genes and other regions in the UCSC browser. The syntax of the bedGraphToBigWig command is the following: "bedGraphToBigWig file.bg $STARgenomeDir/chrNameLength.txt file.bw."

For this particular run, we provide both the distribution of mapped reads into genomic domains in (Fig. 2) and an example of coverage plot over a typical gene, *MYC*, in (Fig. 3).

3.2 Transcriptome Reconstruction/ Assembly

3.2.1 Cufflinks

Cufflinks aims at assembling reads mapped to the genome into transcripts, using or not the annotation as a guide. Therefore it uses as input the bam file generated previously, and optionally the reference annotation in GTF format. Even if cufflinks can also provide quantification of expression of the reconstructed transcripts, here, we will only use it as a transcript modeler (while RSEM will be used for quantification).

3.2.2 Input Files and Arguments

– $nthreads is the number of threads that can be used for the computation.

– $ANNOTATION is the gene annotation in GTF format.

– $outdir is the directory where the results will be stored.

– $libtype is the library type (for illumina stranded or unstranded **Note 9**).

– $bam is the genomic BAM file obtained in the previous step.

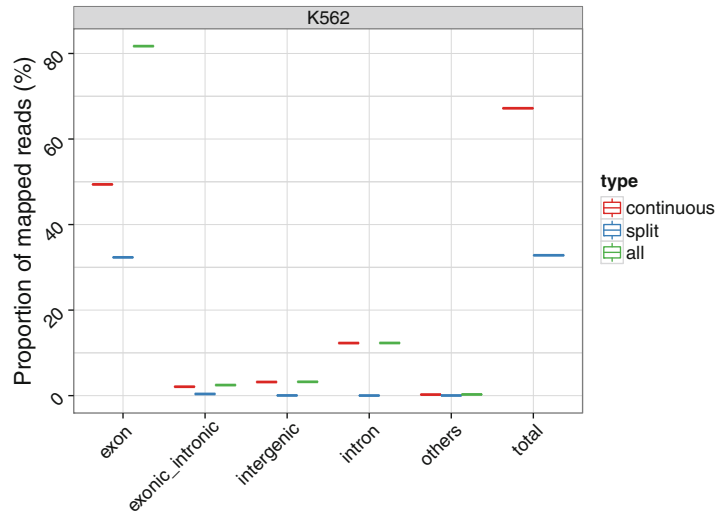

Fig. 2 Mapped read distribution in genomic domains. Primary alignments of reads are partitioned into continuous and split reads and then into the following categories: (1) exonic if they are totally included in exons, (2) intronic if they are totally included in introns, (3) exonic–intronic if they are totally included in genes but not in (1) or (2), (4) intergenic if they are totally included in intergenic regions, and (5) others if they are not in the previous categories. Even if the majority of genes map in a continuous way, the percentage of split-mapped reads is quite high (33 % of the total mapped reads). Most mapped reads fall into exons (82 %), and then introns (12 %), but very few of them lie at exon-intron boundaries and in intergenic regions (<5 %)

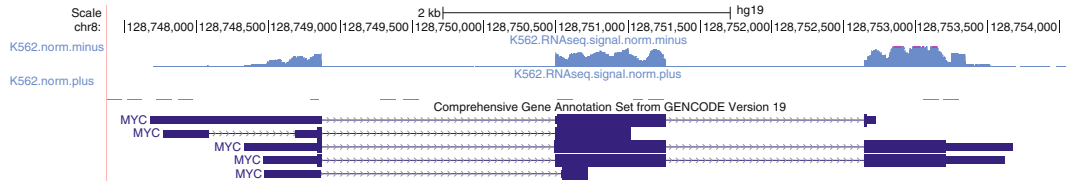

Fig. 3 Read coverage of the MYC gene. This figure shows the + and – strand RNA-seq signal (RNA-seq mapped read aggregation) over the MYC gene in the UCSC browser. As expected from a polyA+ RNA experiment, exons are covered more than introns. Note that this gene has several transcript isoforms annotated in Gencode v19

3.2.3 Command

```
cufflinks -p $nthreads -g $ANNOTATION -o $outdir
-u \
--library-type fr-firststrand $libtype $bam
```

The -g/--GTF-guide option tells cufflinks to use the reference transcript annotation to guide the assembly. Unlike the -G/--GTF option which will ignore alignments that are not present in $ANNOTATION, the -g option allows the identification of novel transcript isoforms.

3.2.4 Output File

$outdir/transcripts.gtf is the file containing the transcript assembly produced by cufflinks.

3.3 Transcript and Gene Quantifications

RSEM uses reads mapped to the transcriptome to quantify the expression of transcripts and genes. It uses an expectation maximization approach to rescue multi-mapped reads based on the location of unique reads in the transcript, in an iterative way that stops when the error made is lower than a threshold.

Preparing the RSEM reference files

This needs to be done only once for a given project. The same reference files will then be used for all RNA-seq datasets of this project.

3.3.1 Input Files and Arguments

- $RSEMgenomeDir is the directory where the RSEM indices will be stored. This directory has to be created by the user before running the command.
- $GENOME is the genome sequence in FASTA format.
- $ANNOTATION is the gene annotation in GTF format.

3.3.2 Command

```
mkdir $RSEMgenomeDir
rsem-prepare-reference --gtf $ANNOTATION
   $GENOME $RSEMgenomeDir/RSEMref
```

3.3.3 Output Files

The above command generates 7 output files starting with the RSEMref prefix in the $RSEMgenomeDir output directory, of which only one is of interest to the user (RSEMref.transcripts.fa) and contains the extracted reference transcripts in Multi-FASTA format. The other ones are either used by RSEM internally (RSEMref.grp, RSEMref.ti, RSEMref.transcripts.fa, RSEMref.seq, RSEMref.chrlist) or useful when mapping is done within RSEM which is not the case here, see --no-bam-output option below (RSEMref.idx.fa and RSEMref.n2g.idx.fa).

3.4 Running the Quantification Process

3.4.1 Input Files and Arguments

- $nThreadsRSEM is the number of threads that can be used for the computation.
- Aligned.toTranscriptome.out.bam is the transcriptome BAM file generated previously by STAR.
- $RSEMgenomeDir is the directory where the RSEM reference files are located.

3.4.2 Command

```
rsem-calculate-expression --bam --no-bam-
output --estimate-rspd \
--calc-ci --seed 12345 -p $nThreadsRSEM --ci-
   memory 30000 --paired-end \
--forward-prob 0 Aligned.toTranscriptome.out.
   bam $RSEMgenomeDir/RSEMref Quant
```

The --bam and --no-bam-output options are used to specify that a transcriptome bam file is provided as input (as opposed to

default FASTQ files), and that the program should not generate a BAM file, respectively. The --estimate-rspd option is used to estimate the read start position distribution (RSPD) from the data, and the --calc-ci option calculates 95% credibility intervals and posterior mean estimates. The --seed 12345 option sets the seed for the random number generators used in calculating posterior mean estimates and credibility intervals, while the --paired-end and the --forward-prob 0 are used to specify that the data is paired-end and stranded with all the first reads coming from the opposite strand of the transcript, respectively. Quant is simply the name of the sample, used to label some output files.

3.4.3 Output Files

The rsem-calculate-expression command generates several output files in the current working directory, described in details in http://deweylab.biostat.wisc.edu/rsem/rsem-calculate-expression.html, the most important of which are the following two:

- Quant.isoforms.results, which is a TSV file containing the expression of the annotated transcripts. The 4 most important columns are 1, 5, 6, and 7 which respectively contain the transcript id, the number of reads assigned to this transcript, and two relative measures of its expression: the TPM (Transcript Per Million) and the FPKM (Fragment Per Kilobase of transcript per Million mapped reads).
- Quant.genes.results, for gene quantifications (with the same kind of file format as for transcript isoforms).

Note that TPM is the native RSEM measure of expression, and should be preferred over FPKM. Indeed while the sum of the FPKMs of all transcripts is not constant across samples, the sum of the TPMs of all transcripts expressed in a given sample is always 1, and therefore constant across samples. These quantification files are very useful for visualizing the distribution of transcript or gene expression in a given sample (Fig. 4), as well as for rapidly extracting the number of transcripts or genes detected in a given experiment (Table 3).

4 Notes

1. Single-end protocol:
   If the protocol is single-end some of the above commands have to be slightly modified:
   - STAR mapping: remove "$read2" in the command.
   - RSEM quantification: remove "--paired-end" from the command.

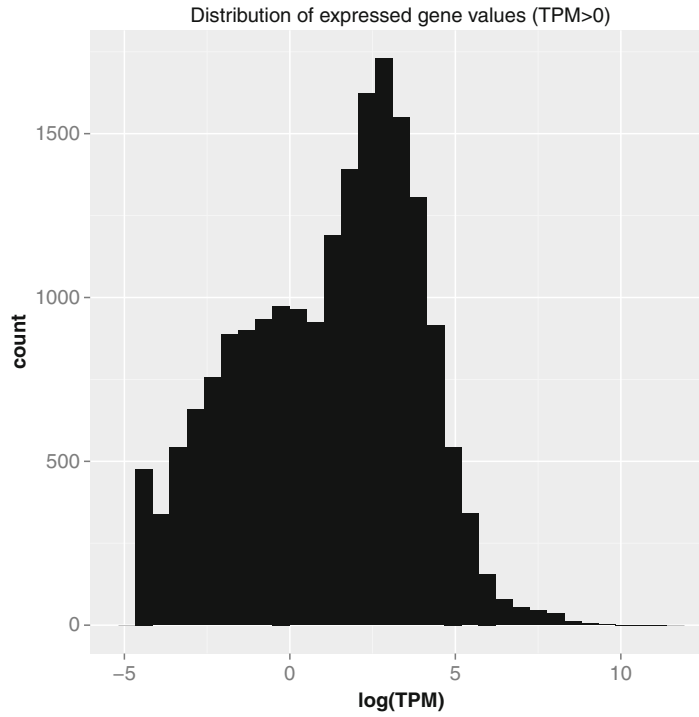

Fig. 4 Gene expression level. Log10 transformed TPM (transcript per million) values of expressed genes is plotted as an histogram. Globally, about 1/3 of the genes are expressed at the threshold of 0 TPM (*see* Table 2)

Table 3

Detected genes and transcripts

		Detected (TPM > 0)			Detected (TPM > 0)	
Super biotype	**# Genes**	**#**	**%**	**# Transcripts**	**#**	**%**
Protein_coding[a]	20,730	14,068	67.9	95,319	43,319	45.4
IncRNA[b]	13,870	3262	23.5	76,684	31,142	40.6
Pseudogene[c]	14,206	1959	13.8	15,343	1929	12.6
SmallRNA[d]	9013	35	0.4	9173	41	0.4
All	57,819	19,324	33.4	196,519	76,431	38.9

Number of annotated and detected genes and transcripts (TPM > 0), for 4 super biotypes (protein_coding, lncRNA, pseudogene, smallRNA), and for all annotated elements. The list of individual Gencode v19 biotypes belonging to each of the 4 super biotypes defined here is indicated at the bottom of the table

[a]IG_C_gene,IG_D_gene,IG_J_gene,IG_V_gene,nonsense_mediated_decay,non_stop_decay,protein_coding,TR_C_gene,TR_D_gene,TR_J_gene,TR_V_gene

[b]3prime_overlapping_ncrna,antisense,lincRNA,processed_transcript,retained_intron,sense_intronic,sense_overlapping

[c]All gencode biotypes containing the term pseudogene

[d]miRNA,misc_RNA,Mt_rRNA,Mt_tRNA,rRNA,snoRNA,snRNA

2. For whole RNA transcriptome sequencing (capturing both polyA+ and polyA- transcripts) it could be of importance to use the unplaced scaffolds when mapping, even if they do not contain any gene, since they act as a sponge of ribosomal reads at the level of the mapping. For instance, the hg19 assembly contains 59 unplaced supercontigs with sizes ranging from 4.2 kb (contig GL000207.1) to 5.5 Mb (for contig GL000207.1) and a total of 6.1 Mb.
3. One primordial task in many HTS bioinformatics analyses is to check whether the chromosome names present in the annotation file correspond to the ones present in the genome file. For instance, genome files from UCSC [18] are not compatible with annotation files from EnsEMBL [23] since the former use the chr_n convention while the latter use the n convention.
4. The command line for exporting the PATH variable depends on the shell language used. In this protocol, we use bash, but for tcsh or csh shells, the syntax should be the following:

```
sentenv PATH $PATH:/path/to/programdir
```

5. Read quality/ QC

   RNA-seq reads deposited in reference databases such as SRA or ENA have generally been precleaned and are normally of good quality, but before using raw sequences generated by sequencing machines, we need to check their quality and possibly clean them to get rid of adapters, contaminants and low quality regions that were introduced in the sequence at various stages of the RNA-seq library preparation.

   Regardless of the sources of the data, the first step consists in assessing read quality. One of the reference tools to this aim is FastQC (http://www.bioinformatics.babraham.ac.uk/projects/fastqc) which provides an HTML report on different quality metrics (phred score, GC%, GC and k-mer bias, duplication, adapter contamination, etc.). It can either be launched on the command line (it is then possible to parallelize the processing for big datasets) or in interactive mode for smaller datasets. For example, for the processing of two files on 8 cores:

```
fastqc -t 8 seqfile1 seqfile2
```

   Then we need to clean the reads. If the cleaning steps are less critical in the case of a genome-guided assembly which use local mapping tools (e.g., Bowtie2/STAR), than in the case of de novo assembly, the ultimate goal is still to assign reads to their correct positions and thus remove low quality regions that potentially contain errors and introduce biases in the quantifications.

   Errors should be removed in the reverse order of the sources that have generated them. (1) First the so-called sequencing-related technical errors: low quality read parts and technical

contaminations like adapters. Trimmomatic (http://www.usadellab.org/cms/?page=trimmomatic) performs an adaptive trimming from the read ends and applies a sliding window over the entire sequence. It preserves information on paired and singletons and performs cleaning from a list of adapters sequences. To use it:

```
trimmomatic PE -threads 8 r1.fq.gz
r2.fq.gz r1_paired.fq.gz r1_unpaired.
fq.gz r2_paired.fq.gz r2_unpaired.fq.gz
ILLUMINACLIP:adaptor_list.fa:2:30:10
LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 min-
len:50
```

Secondly, the so-called biological errors related to the library preparation protocol and contamination: polyA-tails, rRNA sequences, mtDNA sequences. Since mitochondrial and ribosomal RNAs could be polyadenylated, the use of a polyA selection is not an absolute guarantee to get rid of all these molecules that often represent more than 90% of the cellular RNAs. Ribopicker (ribopicker.sourceforge.net/) can be used to identify and remove contaminant from a dataset. It can be used in the following way:

```
ribopicker -f r1_paired.fq.gz -dbs bwa_in-
dexed_rna_db
```

Finally, it could be necessary to run the trimmomatic program again to maintain, after biological contamination cleaning, paired read integrity and remove shorter reads.

6. Pipeline implementation using ENCODE3 shell scripts.

   Stand-alone shell scripts exist for the STAR/RSEM part of the pipeline used here, however the parameters used for mapping are slightly different from the ones described above, corresponding to the ones used in the official ENCODE3 long RNA-seq pipeline. Note that these scripts, partially documented, only quantify annotated genes :

   (a) Making the indices and reference files: https://github.com/ENCODE-DCC/long-RNASeq-pipeline/blob/master/DAC/STAR_RSEM_prep.sh.

   (b) Mapping the reads, making bigwigs, and quantifying annotated genes: https://github.com/ENCODE-DCC/long-RNASeq-pipeline/blob/master/DAC/STAR_RSEM.sh.

7. Pipeline implementation using nextflow

   Nextflow (www.nextflow.io) is a programming language that eases the writing of computational pipelines with complex data. A nextflow implementation of the mapping and quantification parts of the above pipeline, called grape [24], exists and

may be simpler to use than each individual step (although it will only quantify annotated genes). In order for grape to run with STAR for mapping and RSEM for quantification, the "starrsem" profile needs to be used.

8. Using control RNA spike-ins.

   RNA spike-ins are synthetic RNA molecules added to the RNA library in known amounts in order to be able to calibrate the expression measurements of annotated genes (see [25] for an example). In case they are available, it is recommended to use them as additional reference genes, even if current normalization strategies using them do not necessarily perform better than others [26]. This implies a simultaneous mapping the reads to the spike-ins at the same time as to the genome and transcriptome altogether. Similarly, the expression quantification should include the spike-ins in the set of known (or known and novel) genes. This involves slightly modifying some of the above commands:

   – STAR indexing: add $fastaSpikeins after $fastaGenome, where $fastaSpikeins is a FASTA file with the spike-ins, e.g., spikes.fixed.fasta.
   – RSEM reference file generation: replace $fastaGenome by $fastaGenome","$fastaSpikeins.

9. Unstranded protocol:

   If the protocol is unstranded, some of the above commands have to be slightly modified:

   STAR mapping: add --outSAMstrandField intronMotif so the intron motif at the boundary of a split-mapped reads can be used to determine the mapping strand (this option generates the XS strand attribute for all alignments containing a splice junction, and eliminates all the ones with an undefined strand), and replace Stranded by Unstranded in the --outWigStrand option for wiggle file generation.

   Cufflinks reconstruction: replace "--library-type fr-firststrand" by "--library-type fr-unstranded."

   RSEM quantification: remove --forward-prob 0 from the command line.

References

1. Wang Z, Gerstein M, Snyder M (2009) RNA-seq: a revolutionary tool for transcriptomics. Nature 10:57–63
2. Djebali S, Davis CA, Merkel A et al (2012) Landscape of transcription in human cells. Nature 488:101–108

3. Dobin A, Davis CA, Schlesinger F et al (2012) STAR: ultrafast universal RNA-seq aligner. Bioinformatics 29:15–21
4. Trapnell C, Williams BA, Pertea G et al (2010) Transcript assembly and quantification by RNA-seq reveals unannotated transcripts and isoform switching during cell differentiation. Nat Biotechnol 28:511–515
5. Li B, Ruotti V, Stewart RM et al (2010) RNA-seq gene expression estimation with read mapping uncertainty. Bioinformatics 26:493–500
6. T.E.P. Consortium, T.E.P. Consortium, O.C. Data Analysis Coordination et al (2013) An integrated encyclopedia of DNA elements in the human genome. Nature 488:57–74
7. Martens JHA, Stunnenberg HG (2013) BLUEPRINT: mapping human blood cell epigenomes. Haematologica 98:1487–1489
8. Steijger T, Abril JF, Engström PG et al (2013) Assessment of transcript reconstruction methods for RNA-seq. Nat Methods 10:1177–1184
9. Engström PG, Steijger T, Sipos B et al (2013) Systematic evaluation of spliced alignment programs for RNA-seq data. Nat Methods 10: 1185–1191
10. Roberts A, Goff L, Pertea G et al (2012) Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks. Nat Protoc 7:562–578
11. Marco-Sola S, Sammeth M, Guigó R et al (2012) The GEM mapper: fast, accurate and versatile alignment by filtration. Nat Methods 9:1185–1188
12. Pertea M, Pertea GM, Antonescu CM et al (2015) StringTie enables improved reconstruction of a transcriptome from RNA-seq reads. Nat Biotechnol 33:290–295
13. Montgomery SB, Sammeth M, Gutierrez-Arcelus M et al (2010) Transcriptome genetics using second generation sequencing in a Caucasian population. Nature 464:773–777
14. Roberts A, Pachter L (2013) Streaming fragment assignment for real-time analysis of sequencing experiments. Nat Methods 10:71–73
15. Patro R, Mount SM, Kingsford C (2014) Sailfish enables alignment-free isoform quantification from RNA-seq reads using lightweight algorithms. Nat Biotechnol 32:462–464
16. Haas BJ, Papanicolaou A, Yassour M et al (2013) De novo transcript sequence reconstruction from RNA-seq using the Trinity platform for reference generation and analysis. Nat Protoc 8:1494–1512
17. Sacomoto GAT, Kielbassa J, Chikhi R et al (2012) KISSPLICE: de-novo calling alternative splicing events from RNA-seq data. BMC Bioinformatics 13(Suppl 6):S5
18. Rosenbloom KR, Sloan CA, Malladi VS et al (2013) ENCODE data in the UCSC Genome Browser: year 5 update. Nucleic Acids Res 41:D56–D63
19. Harrow J, Frankish A, Gonzalez JM et al (2012) GENCODE: the reference human genome annotation for The ENCODE Project. Genome Res 22:1760–1774
20. Derrien T, Johnson R, Bussotti G et al (2012) The GENCODE v7 catalog of human long noncoding RNAs: analysis of their gene structure, evolution, and expression. Genome Res 22:1775–1789
21. Pei B, Sisu C, Frankish A et al (2012) The GENCODE pseudogene resource. Genome Biol 13:R51
22. Li H, Handsaker B, Wysoker A et al (2009) The sequence alignment/map format and SAMtools. Bioinformatics 25:2078–2079
23. Cunningham F, Amode MR, Barrell D et al (2015) Ensembl 2015. Nucleic Acids Res 43:D662–D669
24. Knowles DG, Röder M, Merkel A et al (2013) Grape RNA-seq analysis pipeline environment. Bioinformatics 29:614–621
25. Jiang L, Schlesinger F, Davis CA et al (2011) Synthetic spike-in standards for RNA-seq experiments. Genome Res 21:1543–1551
26. Risso D, Ngai J, Speed TP et al (2014) Normalization of RNA-seq data using factor analysis of control genes or samples. Nat Biotechnol 32:896–902