TruSeq Ribo Profile

Bioinformatics Guide

FOR RESEARCH USE ONLY

Introduction

This guide provides suggestions for analyzing data analysis from libraries prepared using TruSeq Ribo Profile Kits and sequenced on an Illumina sequencing system.

NOTE
While this document provides instructions on the analysis of ribosomal profiling samples, it is not intended to be a comprehensive guide. Also note that TopHat is not an Illumina-supported product.

The workflow described in this guide assumes that you have command-line access to a UNIX server with 16 GB of RAM (64 GB RAM preferred), 64-bit architecture, and basic knowledge of UNIX. In addition, you may require assistance from your IT department for software installation and permissions.

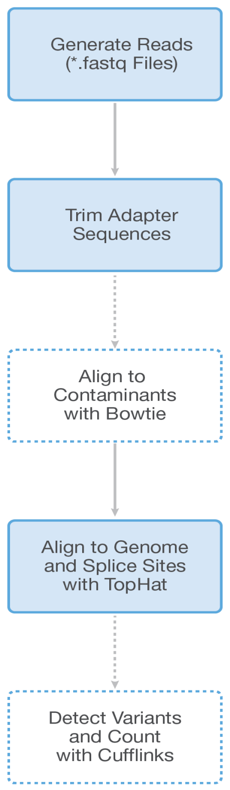
Workflow Overview

There are 4 main steps in the TruSeq Ribo Profile analysis pipeline:

1. Generation of FASTQ files
2. Trimming adapter sequences
3. Alignment to remove contaminants (optional)
4. Alignment to a reference genome and splice junctions using TopHat

After alignment, the resulting BAM file can be used for downstream analyses with other bioinformatics tools. This guide demonstrates the use of the Cufflinks package[1] for transcript annotation, assembly, quantification, and visualization.

Figure 1 Overview of the TruSeq Ribo Profile Analysis Pipeline



Optional steps are indicated with dashed lines.

Software Installation

Information and guides to install the required tools:

- CASAVA (http://support.illumina.com/sequencing/sequencing_software/casava/questions.ilmn)
- Bowtie (http://bowtie-bio.sourceforge.net)
- TopHat (http://TopHat.cbcb.umd.edu/)
- Cufflinks (http://cufflinks.cbcb.umd.edu/)
- SAMtools (http://samtools.sourceforge.net)
- FASTX toolkit (http://hannonlab.cshl.edu/fastx_toolkit/index.html)

These versions were used in the development of the TruSeq Ribo Profile analysis workflow:

- CASAVA (v1.8)
- Bowtie (v0.12.7)
- TopHat (v1.4.1)
- Cufflinks (v1.3.0)
- SAMtools (v0.1.18)
- FASTX toolkit (v0.0.13.2)

Though more recent versions of the programs will also be compatible with this pipeline, the workflow is intended to function with the versions listed.

For further installation instructions, refer to the Appendix, on page 27 or the *Getting Started* or *Download* link for each tool listed.

Once the tools have been installed, you will need to make sure that the UNIX environment variables are appropriately set. You can either add the location of the executables installed to your PATH variable or create a new directory called bin in your home directory, copy the executables to this location, and add the location of the bin directory to your PATH variable.

To change your PATH variable, enter (assuming bash shell):

```
> export PATH = <list of paths>:$PATH
```

Testing the Installation

Bowtie

```
>bowtie --version
```

This command should return output similar to the following:

```
bowtie version 0.12.7 64-bit
Built on sd-qmaster.illumina.command-line Mon Feb 8 14:03:53 PST
2010
Compiler: gcc version 4.1.2 20080704 (Red Hat 4.1.2-44)
Options: -O3
Sizeof {int, long, long long, void*, size_t, off_t}:
{4, 8, 8, 8, 8, 8}
```

TopHat

```
>TopHat --version
```

This command should return output similar to the following:

```
TopHat v1.4.1
…
```

Cufflinks

```
>cufflinks
```

This command should return output similar to the following:

```
Cufflinks v1.3.0
…
```

FASTX toolbox

```
>fastx_trimmer -h
```

This command should return output similar to the following:

```
usage: fastx_trimmer [-h] [-f N] [-l N] [-t N] [-m MINLEN] [-z]
[-v] [-i INFILE] [-o OUTFILE] Part of FASTX Toolkit 0.0.13.2 by A.
Gordon (gordon@cshl.edu)
[-h] = This helpful help screen.
…
```

Verify that there are no errors. If any of these commands generates an error, check your installation.

Genome Reference Files

In order to align libraries, you will need to download the desired genome. Illumina provides these resources in the iGenomes repository for the following organisms:

- Arabidopsis_thaliana
- Bos_taurus
- Caenorhabditis_elegans
- Canis_familiaris
- Drosophila_melanogaster
- Equus_caballus
- Escherichia_coli_K_12_DH10B
- Escherichia_coli_K_12_MG1655
- Gallus_gallus
- Homo_sapiens
- Mus_musculus
- Mycobacterium_tuberculosis_H37RV
- Pan_troglodytes
- PhiX
- Rattus_norvegicus
- Saccharomyces_cerevisiae
- Sus_scrofa

The iGenomes repository can be accessed from the Illumina FTP site: ftp://ussd-ftp.illumina.com

Example

1. Download the genome sequences for the human hg19 build from the iGenomes repository with the following commands:
   ```
   > wget --ftp-user=igenome --ftp-password=G3nom3s4u ftp://ussd-
   ftp.illumina.com/Homo_sapiens/UCSC/hg19/Homo_sapiens_UCSC_
   hg19.tar.gz
   ```
2. Log in using the following credentials:
   a. Username: igenome
   b. Password: G3nom3s4u
3. Unpack the downloaded file using the following command:
   ```
   > tar -xvzf Homo_sapiens_UCSC_hg19.tar.gz
   ```
4. Unpacking will make its own folder:
   ```
   Homo_sapiens/UCSC/hg19
   ```
5. Within this folder, you will find the following subfolders:
   ```
   Homo_sapiens/UCSC/hg19/Annotation/Genes/genes.gtf
   Homo_sapiens/UCSC/hg19/Sequence/BowtieIndex/genome*.*
   ```
   or Bowtie2 indices at
   ```
   Homo_sapiens/UCSC/hg19/Sequence/Bowtie2Index/genome*.*
   ```

Test Samples

Illumina provides 3 samples along with this guide. In addition, the README file indicates what these samples are and includes a list of commands from this guide.

- http://www.epibio.com/artseq_samples/README.txt
- http://www.epibio.com/artseq_samples/ArtSeq.sh
- http://www.epibio.com/artseq_samples/494_5_NoIndex_L005_R1_.fastq.gz
- http://www.epibio.com/artseq_samples/494_10_NoIndex_L002_R1_.fastq.gz
- http://www.epibio.com/artseq_samples/494_11_NoIndex_L003_R1_.fastq.gz

These samples are FASTQ files containing data from:

1. TruSeq Ribo Profile library from RPFs purified using size-exclusion chromatography (SEC)
2. TruSeq Ribo Profile library from RPFs purified using sucrose cushion
3. Control library from total RNA

Analysis Workflow

Setting up a Workflow Directory

Before starting the alignment, create a directory to contain the results of this workflow:

```
>mkdir <WorkflowFolder>
>cd <WorkflowFolder>
```

Parameter	Description
`<WorkflowFolder>`	Path and folder in which to store the results.

Converting BaseCalls

The standard sequencing output for the HiSeq, Genome Analyzer, and RTA v1.13 consists of *.bcl files, which contain the base calls and quality scores per cycle. TopHat uses *.fastq.gz files as input. To convert *.bcl files into *.fastq.gz files, use CASAVA v1.8. In addition to generating FASTQ files, CASAVA uses a user-created sample sheet to divide the run output in projects and samples, and stores them in separate directories. Each directory can be analyzed independently and contains the files necessary for alignment, variant analysis, and counting with CASAVA.

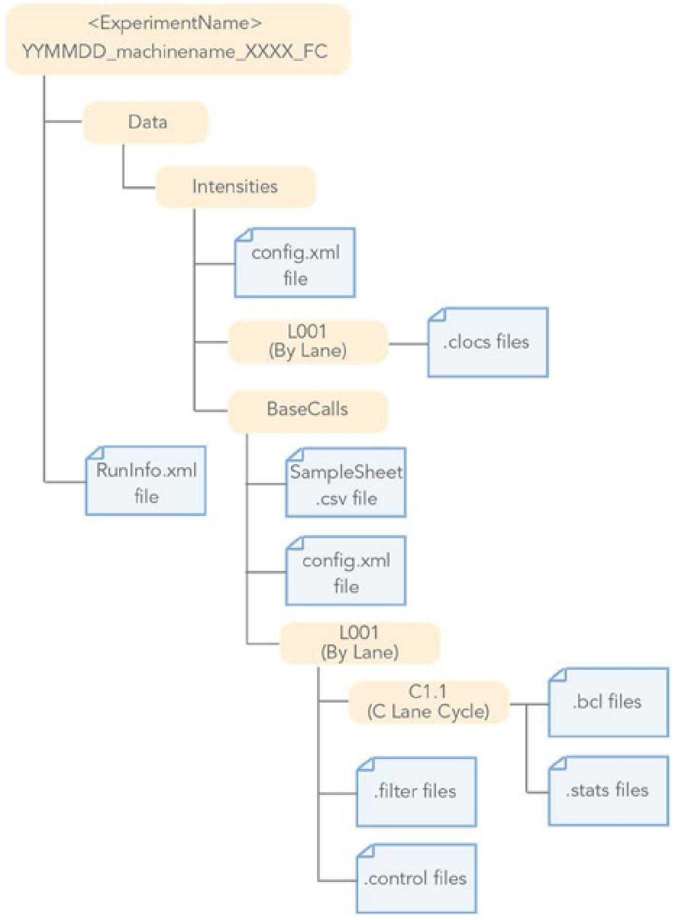
At the same time, CASAVA also separates multiplexed samples (demultiplexing). Multiplexed sequencing allows you to run multiple individual samples in 1 lane. The samples are identified by index sequences that were attached to the template during sample prep. The multiplexed samples are assigned to projects and samples based on the sample sheet, and stored in corresponding project and sample directories.

NOTE
For a comprehensive description of bcl conversion, see the CASAVA v1.8 manual at http://support.illumina.com/sequencing/sequencing_software/casava/documentation.ilmn.

Bcl Conversion Input Files

Demultiplexing requires a BaseCalls directory and a sample sheet to start a run. These files are depicted in Figure 2.

Figure 2 Folder Structure Required for the CASAVA Demultiplex Program



BaseCalls Directory

Demultiplexing requires a BaseCalls directory as generated by RTA, which contains the binary base call files (*.bcl files).

The BCL to FASTQ converter needs the following input files from the BaseCalls directory:

- *.bcl files
- *.stats files
- *.filter files
- *.control files
- *.clocs, *.locs, or *_pos.txt files. The BCL to FASTQ converter determines which type of position file to look for, based on the RTA version that was used to generate the file.
- RunInfo.xml file. This file is located at the top level of the run folder.
- config.xml file

RTA is configured to copy these files off the instrument computer machine to the BaseCalls directory on the analysis server.

Generating the Sample Sheet

The user-generated sample sheet (SampleSheet.csv file) describes the samples and projects in each lane, including the indexes used. The sample sheet should be located in the BaseCalls directory of the run folder. You can create, open, and edit the sample sheet in Excel. You can generate it using Excel or another text editing tool that allows saving *.csv files. Enter the columns specified in Table 1 for each sample, and save the Excel file in the *.csv format. If the sample you want to specify does not have an index sequence, leave the **Index** field empty.

Table 1 Sample Sheet Columns

Column	Description
FCID	Flow cell ID
Lane	Positive integer, indicating the lane number (1-8)
SampleID	ID of the sample
SampleRef	The reference used for alignment for the sample
Index	Index sequences. Multiple index reads are separated by a hyphen (eg, ACCAGTAA-GGACATGA)
Description	Description of the sample
Control	Y indicates this lane is a control lane, N indicates sample lanes
Recipe	Recipe used during sequencing
Operator	Name or ID of the operator
SampleProject	The project to which the sample belongs

Illegal Characters

Project and sample names in the sample sheet cannot contain illegal characters not allowed by some file systems. The characters not allowed are the space character and the following:

```
? ( ) [ ] / \ = + < > : ; " ' , * ^ | & .
```

Multiple Index Reads

If multiple index reads were used, each sample must be associated with an index sequence for each index read. All index sequences are specified in the **Index** field. The individual index read sequences are separated with a hyphen character (-). For example, if a particular sample was associated with the sequence ACCAGTAA in the first index read, and the sequence GGACATGA in the second index read, the index entry would be ACCAGTAA-GGACATGA.

Samples Without an Index

As of CASAVA 1.8, you can assign samples without an index to projects, sampleIDs, or other identifiers by leaving the **Index** field empty.

Running Bcl Conversion and Demultiplexing

Bcl conversion and demultiplexing is performed by a single script, configureBclToFastq.pl. This section describes how to perform bcl conversion and demultiplexing in CASAVA 1.8.

Usage

The standard method for running bcl conversion and demultiplexing is to create the necessary makefiles first, which configure the run. Then run `make` on the generated files, which executes the commands.

1. Enter the following command to create a makefile for demultiplexing:
   `><path-to-CASAVA>/bin/configureBclToFastq.pl[options]`
2. Move into the newly created Unaligned folder specified by `--output-dir`.
3. Type the `make` command:
   `> nohup make -j N &`
   The optional `&` tells the system to run the analysis in the background, leaving you free to enter more commands. Illumina suggests always running `nohup` to help troubleshooting if issues arise.
   The `-j` option specifies the extent of parallelization, with the options depending on the setup of your computer or computing cluster.
4. After the analysis is complete, review the results for each sample.

Options

The options for demultiplexing are described in Table 2.

Table 2 Options for Demultiplexing

Option	Description	Examples
`--fastq-cluster-count`	Maximum number of clusters per output FASTQ file. Do not go over 16000000, since this is the maximum number of reads we recommend for one ELAND process. Specify 0 to ensure creation of a single FASTQ file. Defaults to 4000000.	`--fastq-clustercount 6000000`
`-i, --input-dir`	Path to a BaseCalls directory.\ Defaults to current dir	`--input-dir <BaseCalls_dir>`
`-o, --output-dir`	Path to demultiplexed output. Defaults to `<run_folder>/Unaligned` Note that there can be only one Unaligned directory by default. If you want multiple Unaligned directories, you will have to use this option to generate a different output directory.	`--output-dir <run_folder>/Unaligned`
`--positions-dir`	Path to a directory containing positions files. Defaults depends on the RTA version that is detected.	`--positions-dir <positions_dir>`

Option	Description	Examples
`--positions-format`	Format of the input cluster positions information. Options: • `.locs` • `.clocs` • `_pos.txt` Defaults to .clocs.	`--positions-format .locs`
`--filter-dir`	Path to a directory containing filter files. Defaults depends on RTA version that is detected.	`--filter-dir <filter_dir>`
`--intensities-dir`	Path to a valid Intensities directory. Defaults to parent of `base_calls_dir`.	`--intensities-dir <intensities_dir>`
`-s,--sample-sheet`	Path to sample sheet file. Defaults to `<input_dir>/SampleSheet.csv`	`--sample-sheet <input_dir>/SampleSheet.csv`
`--tiles`	The `--tiles` option takes a comma-separated list of regular expressions to match against the expected `s_<lane>_<tile>` pattern, where `<lane>` is the lane number (1–8) and `<tile>` is the 4 digit tile number (left-padded with 0s).	`--tiles=s_[2468]_[0-9][0-9][02468]5,s_1_0001`
`--use-bases-mask`	The `--use-bases-mask` string specifies how to use each cycle. • An `n` means ignore the cycle. • A `Y` (or `y`) means use the cycle. • An `I` means use the cycle for the index read. • A number means that the previous character is repeated that many times. • The read masks are separated by commas `,` The format for dual indexing is as follows: `-- use-bases-mask Y*,I*,I*,Y*` or variations thereof as specified above. If this option is not specified, the mask will be determined from the RunInfo.xml file in the run directory. If it cannot do this, you will have to supply the `--use-bases-mask`.	`--use-bases-mask y50n,I6n,Y50n` This means: • Use first 50 bases for first read (Y50) • Ignore the next (n) • Use 6 bases for index (I6) • Ignore next (n) • Use 50 bases for second read (Y50) • Ignore next (n)
`--no-eamss`	Disable the masking of the quality values with the Read Segment Quality control metric filter.	`--no-eamss`

Option	Description	Examples
`--mismatches`	Comma-delimited list of number of mismatches allowed for each read (for example: `1,1`). If a single value is provide, all index reads will allow the same number mismatches. Default is 0.	`--mismatches 1`
`--flowcell-id`	Use the specified string as the flowcell ID. (Default value is parsed from the config-file.)	`--flowcell-id flow_ cell_id`
`--ignore-missing-stats`	Fill in with zeros when *.stats files are missing	`--ignore-missing-stats`
`--ignore-missing-bcl`	Interpret missing *.bcl files as no call	`--ignore-missing-bcl`
`--ignore-missing- control`	Interpret missing control files as not-set control bits	`--ignore-missing-control`
`--with-failed-reads`	Include failed reads into the FASTQ files (by default, only reads passing filter are included).	`--with-failed-reads`
`--adapter-sequence`	Path to a FASTA adapter sequence file. If there are 2 adapter sequences specified in the FASTA file, the second adapter will be used to mask read 2. Else, the same adapter will be used for all reads. Default: None (no masking)	`--adapter-sequence <adapter dir>/adapter.fa`
`--man`	Print a manual page for this command	`--man`
`-h, --help`	Produce help message and exit	`-h`

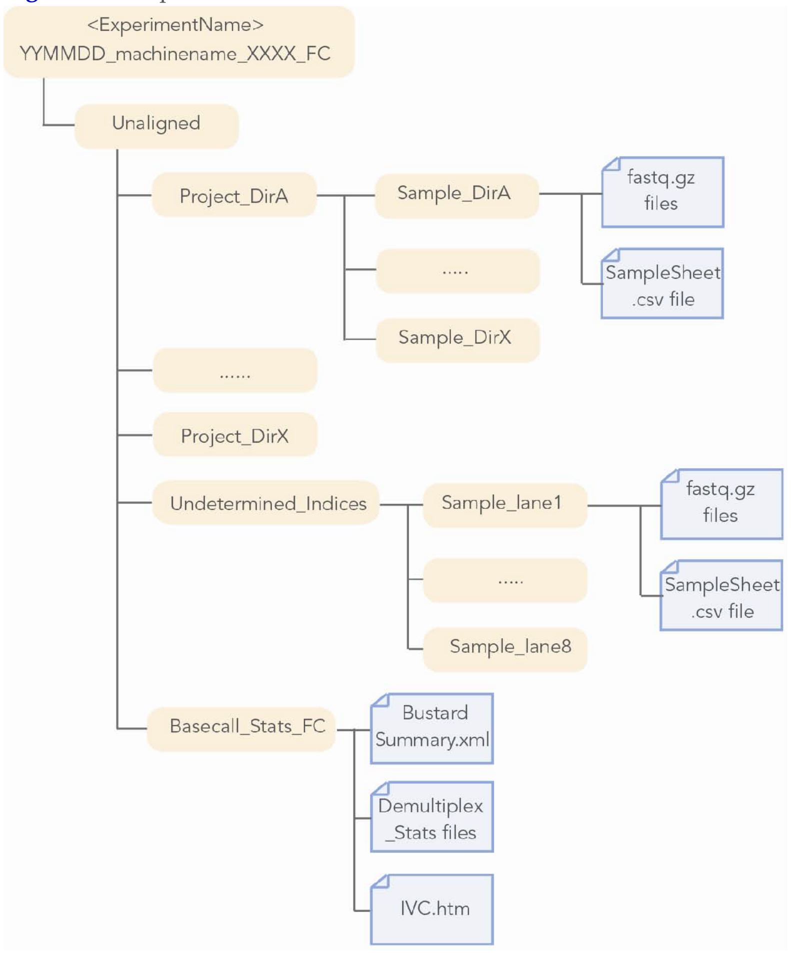
Output

The directory structure that results from bcl conversion output is shown in Figure 3. The output directory has the following characteristics:

- The project and sample directory names are derived from the sample sheet.
- The Demultiplex_Stats file shows where the sample data are saved in the directory structure.
- The Undetermined_Indices directory contains reads with an unresolved or erroneous index.
- If no sample sheet exists, the software generates a project directory named after the flow cell and sample directories for each lane.
- Each directory is a valid base calls directory that can be used for subsequent alignment analysis.

NOTE
If the majority of reads end up in the Undetermined_Indices folder, check the `--use-bases-mask` parameter syntax and the length of the index in the sample sheet. Set the `--use-bases-mask` option to the length of the index in the sample sheet + the character `n` to account for phasing. You will not be able to see which indices have been placed in the `Undetermined_Indices` folder.

Figure 3 Output Folder Structure



Using FASTX Toolbox

Read Preprocessing and Adapter Removal

Due to the short length of ribosome-protected fragments (RPFs) relative to read lengths, most reads from TruSeq Ribo Profile libraries will contain 3′ adapter sequences. For compatibility with the Bowtie and TopHat aligners, the adapter sequences need to be trimmed from the 3′ end of the read. Illumina recommends the use of the FastX toolkit. The tool can be downloaded from http://hannonlab.cshl.edu/fastx_toolkit/download.html.

The Appendix, on page 27 contains further instructions on installing the FastX toolbox.

To trim reads using the FastX toolbox, change directory to the location of workflow folder and run the following commands:

```
> cd <Workflow Folder>
> mkdir -p ./Trimmed/< sampleID>
> zcat <Path To Fastq>/*.fastq.gz | fastx_clipper -a <adapter
sequence> -l <length> -c -n -v -Q33 | fastx_trimmer -Q33 -f <cycle
to begin alignment> 2>> <sampleID>.log >
./Trimmed/<sampleID>/trimmed.fastq
```

Parameter	Definition
`<Path To Fastq>`	The location of the FASTQ files generated during bcl conversion, including the Project and Sample names, eg, <my run folder>/Unaligned/Project_myProject/Sample6.
`<adapter sequence>`	The adapter sequence used in library preparation. For TruSeq Ribo Profile libraries, this sequence is AGATCGGAAGAGCACACGTCT.
`<sampleID>`	The name of the sample being converted.
`<length>`	The minimum length of read to maintain after trimming the adapter sequence. The default value is 5, but Illumina recommends a value of 20–25, depending on the target footprint length.
`<cycle to begin alignment>`	The number of the first cycle to consider for alignment, adjusted to account for any 5′-specific errors, eg, nontemplated additions. This value can be adjusted based on estimated error rates in the first few cycles of the run. The currently recommended value for this parameter is 1.

Further explanation of command line options for the fastx_trimmer and fastx_clipper is available on the Hannon lab website at http://hannonlab.cshl.edu/fastx_toolkit/commandline.html.

The output from this command will be a fastq file for the sample with adapter sequence trimmed off. The data output will be in the ./Trimmed/<sampleID> folder.

Example

The following example uses the commands for the human sample test data:

```
>zcat Unaligned/Project_human/Sample_494_5/*fastq.gz | fastx_
clipper -a AGATCGGAAGAGCACACGTCT -l 25 -c -n -v -Q33 | fastx_
trimmer -Q33 -f 1 2>>Sample_494_5.log > ./Trimmed/Sample_494_
5/trimmed.fastq
```

Running the command produces the file ./Trimmed/Sample_494_5/trimmed.fastq

The output from the trimmer program removes 9.5% of the input reads and results in the following output:

```
Clipping Adapter: AGATCGGAAGAGCACACGTCT
Min. Length: 25
Non-Clipped reads - discarded.
Input: 65928633 reads.
Output: 62917881 reads.
discarded 2327300 too-short reads.
discarded 506083 adapter-only reads.
discarded 177369 non-clipped reads.
```

Contaminant Removal

In order to remove and quantify ribosomal RNA (rRNA) content or other contaminants in your sample prior to alignment to the genome, you can align the trimmed reads against specific contaminant sequences. Alternatively, you can skip these steps at this stage and mask out contaminating sequences masked after alignment.

The first step in removing contaminants is to create a FASTA formatted file containing contaminating sequences from your sample to align against, using the Bowtie aligner.[2]

Preparation of FASTA File

A list of common contaminants, including rRNA and mitochondrial RNA sequences, is available via the iGenomes server for selected organisms listed in the iGenomes directory. For example, the rRNA sequence for *Homo sapiens* can be found in your installation of the iGenomes directory in:

```
<your path to iGenomes>/Homo_sapiens/UCSC/hg19/
Sequence/AbundantSequences/hum5SrDNA.fa
```

and

```
<your path to iGenomes>/Homo_sapiens/UCSC/hg19/
Sequence/AbundantSequences/humRibosomal.fa
```

Similarly, the ribosomal sequences for yeast are found in the file: RDN18-1.fa, RDN25-1.fa, RDN37-1.fa, and RDN58-1.fa, located in the iGenomes repository at:

```
<your path to iGenomes>/Saccharomyces_cerevisiae/UCSC/sacCer3/
Sequence/AbundantSequences/
```

If you have your own sequences you wish to align against, copy and paste your sequences in FASTA format. For example,

```
>gi|555853|gb|U13369.1|HSU13369 Human ribosomal DNA complete
repeating unit
GCTGACACGCTGTCCTCTGGCGACCTGTCGTCGGAGAGGTTGGGCCTCCGGATGCGCGCGGGGCT
CTGGC
CTCACGGTGACCGGCTAGCCGGCCGCGCTCCTGCCTTGAGCCGCCTGCCGCGGCCCGCGGGCCTG
CTGTT......
```

To make the contaminant sequences compatible with Bowtie, run the `bowtie-build` command on the contaminant FASTA file generated above to create special indexed files, called EBWT files. The command-line arguments are:

```
> bowtie-build <comma separated list of fastaFiles to include>
<indexName>
```

Parameter	Description
`<indexName>`	Name given to the EBWT files.

Example

This example demonstrates creating a directory containing EBWT ribosomal sequences in an output directory contam/rRNA with the index name rRNA:

```
> mkdir -p contam/rRNA
> cd contam/rRNA
> bowtie-build <path to file>/humRibosomal.fa,<path to
file>/hum5SrDNA.fa rRNA
```

rRNA Alignment with Bowtie

The next step in the process is to align the trimmed reads to the ribosomal EBWT files generated previously.

To account for the short read-length values generated by RPFs, default Bowtie parameters have been modified by specifying a smaller seed length value of 20, compared to the default value of 28 for longer reads. For more information on alignment parameters, see http://bowtie-bio.sourceforge.net/manual.shtml.

The aligned reads will be written to the file `<outfile name>` and the unaligned reads will be written to `<unalignedFastFileName>`:

```
> bowtie -l 20 --un=<unalignedFastqFileName> <path to ebwt rRNA
folder>/<indexName> <trimmed fastq file> 2>> stats.txt > <outfile
name>
```

Parameter	Definition
`<path to ebwt rRNA folder>`	The folder generated with EBWT files for contaminants.
`<indexName>`	The name of the EBWT files.
`<trimmed fastq file>`	The FASTQ file generated with clipped sequences.
`<outfile name>`	The name assigned to the file to contain the rRNA sequence alignments.

Example

After the analysis for Sample_5 of the test data, enter the following command:

```
> bowtie -l 20 --un=norrna.fastq contam/rRNA/rRNA
./Trimmed/Sample_494_5/trimmed.fastq 2>> stats.txt >
rrnaAlignments.aln
```

The alignment summary (number of input sequences and the number of reads aligned to rRNA sequences) is contained in the file stats.txt.

Removing Additional Contaminant Sequences

In addition to rRNA, your sample can have other contaminating sequences, such as transfer (tRNA), that you wish to quantify and remove prior to alignment to the genome and splice junctions. Due to the compact nature, size (~75 nt), and stable structure of tRNAs, RNase I digestion can cleave the individual molecules in half. This results in two fragments that are roughly similar in size to RPFs and can thus become a major contaminant in the samples. The use of sufficiently high levels of ribonuclease can overcome this problem, although a significant fraction of tRNA contamination can remain.

For tRNA alignments, you can simply combine tRNA sequences with the rRNA sequences from the previous step and run a single contaminant removal analysis. Alternatively, you can choose to run another iteration of Bowtie alignment against tRNA sequences, to quantify tRNA contamination separately from rRNA.

Similar to the rRNA removal, the steps involved in aligning to these sequences are:

1. Prepare a FASTA file of the contaminants. For example, if you have obtained a list of tRNA sequences, then create a file called tRNA.fa or combine these sequences with existing contaminant sequences. The sequences for tRNA are currently not included in the Illumina iGenomes databases. Download them from sources such as the Genomic tRNA database (http://gtrnadb.ucsc.edu/download.html) or Ensembl (http://www.ensembl.org).

2. Create a Bowtie index of your file (EBWT files):
   ```
   > mkdir tRNA
   > bowtie-build <path to file>/tRNA.fa tRNA
   ```

3. Run a Bowtie alignment. Specify the name of the file containing unaligned sequences you wish to use for further downstream analyses, the file name containing reads aligned to the contaminant, and the input sequence reads. For example, to align the reads after rRNA removal to tRNA sequences, run the following alignment:

```
> bowtie -l 20 --un=<unalignedSequences> <path to tRNA ebwt
files>/<indexName> <input fastq file> 2>> stats.txt > <trna
aligned sequences>
```

Parameter	Description
`<unalignedSequences>`	Name of the file containing the unaligned sequences.
`<input fastq file>`	Name of the input FASTQ file.
`<indexName>`	Name of the EBWT files.
`<aligned sequences>`	Name of the file to which the alignments will be written.

Alignment stats from Bowtie will be appended to the file stats.txt.

Example

After the analysis for Sample_5 of the test data, enter the following command:

```
> bowtie -l 20 --un=./Trimmed/Sample_494_5/nocontam.fastq
contam/tRNA/tRNA norrna.fastq 2>> stats.txt > trnaAlignments.aln
```

Sequence Alignment

The next step for analysis is to align the reads to the genome and splice junctions using the TopHat aligner.[2]

Start the single-read alignments for each sample by entering the following command:

```
>topHat --GTF <iGenomesFolder>/Annotation/Genes/genes.gtf --num-
threads 1 --output-dir <SampleOutputFolder>
<iGenomesFolder>/Sequence/BowtieIndex/genome <SampleID
input>.fastq
```

Parameter	Description
`<SampleOutputFolder>`	Path and folder to store the sample output.
`<iGenomesFolder>`	Path and folder of the iGenomes directory.
`<input fastq>`	Name of the FASTQ file to submit for alignment to the genome. This is the same file as the one generated after the contaminant removal step (Removing Additional Contaminant Sequences, on page 16).

NOTE
Illumina also suggests running TopHat with the `--no-novel-juncs` flag specified in the command above to save time if you are not interested in doing novel splice site discovery.

The main option to modify the analysis is the following:

```
--num-threads 1
```

If the workflow is run on a machine with multiple cores, this number may be increased to reflect the number of cores present.

Example

The following example uses the commands for the human sample test data:

```
>mkdir -p ./TopHatOut/Sample_494_5
>topHat --GTFiGenomes/Homo_
sapiens/UCSC/hg19/Annotation/Genes/genes.gtf
--num-threads 8
--output-dir=./TopHatOut/Sample_494_5 iGenomes/Homo_
sapiens/UCSC/hg19/Sequence/BowtieIndex/genome nocontam.fastq
```

TopHat Output Description

After analysis with TopHat, the output directory specified in the run command should contain the following files:

- accepted_hits.bam—This file details the mapping of each read to the genome.
- junctions.bed—This file contains information, including genomic location and supporting evidence, about all of the splice junctions discovered by TopHat in the process of aligning reads.

The accepted_hits.bam file is used for further analysis in quantification of transcripts within the sample. It is compatible with many other third-party bioinformatics tools.

NOTE
For more information about the BAM format, see http://samtools.sourceforge.net.

Detecting and Counting Transcripts (Optional)

At this stage, the SAM file from TopHat only contains the mapped locations of the input reads. Cufflinks is a suite of tools for quantifying aligned RNA sequencing data. The Cufflinks suite assembles these reads into transcripts, as well as quantifies these transcripts in single or multiple experiments.

The Cufflinks suite includes 3 separate tools:

- Cufflinks—assembles novel transcripts and quantifies transcripts against a given annotation.
- Cuffmerge—takes novel transcripts from multiple experiments and combines them into one annotation file.
- Cuffdiff—calculates differential expression given an annotation file; does not need or use the quantification information from Cufflinks.

This guide focuses only on gene/transcript quantification using a known reference genome and differential expression across samples. If using the Cufflinks suite for tasks such as novel junction discovery, see Trapnell et al.[3]

Running Cufflinks

If CuffLinks is provided a reference genome annotation (GTF) file, it will quantify the genes and transcripts specified in that annotation, ignoring any alignments that are incompatible with those annotations. Use the following command to run Cufflinks with an annotation file for all samples:

```
>cd <SampleOutputFolder>
>cufflinks --num-threads 1 -G
<iGenomesFolder>/Annotation/Genes/genes.gtf accepted_hits.bam
>cd ..
```

Parameter	Description
`<SampleOutputFolder>`	Path and folder where the TopHat output is located.
`<iGenomesFolder>`	Path and folder where the genome is stored

WARNING
If you run Cufflinks in the directory where there is already output from another Cufflinks session, it will be overwritten. If you want to keep different Cufflinks outputs, run each Cufflinks session from a different directory, because Cufflinks always prints output to the current directory.

The main options to modify the analysis are the following:

```
--num-threads 1
```

If the workflow is run on a machine with multiple cores, this number may be increased to reflect the number of cores present.

```
-G
```

Tells Cufflinks to use the supplied reference annotation to estimate isoform expression. It will not assemble novel transcripts, and the program will ignore alignments not structurally compatible with any reference transcript.

Example

The following example uses the commands for the human sample test data:

```
>cd ./TopHatOut/Sample_494_5
>cufflinks --num-threads 1 -G
<iGenomesFolder>/Annotation/Genes/genes.gtf accepted_hits.bam
```

If you have multiple samples and conditions, you can get more accurate quantification estimates by using the CuffMerge function from the Cufflinks suite. Instructions on running CuffMerge can be found in Trapnell et al.[3]

Quantification Output

Each output directory contains 2 output files:

- genes.fpkm_tracking: quantifying the expression of genes specified in the GTF annotation file.
- isoforms.fpkm_tracking: quantifying the expression of transcripts specified in the GTF annotation file.

Expression is reported in fragments per kilobase of sequence per million mapped reads (FPKM). This measure normalizes the number of aligned reads by the size of the sequence feature and the total number of mapped reads.

Running Cuffdiff

Cuffdiff performs differential expression analysis between 2 samples on annotated genes. Run Cuffdiff, providing a GTF file and the 2 SAM alignment files, using the following command:

```
>cuffdiff -L <Sample list> -o <OutputDir>
<iGenomesFolder>/Annotation/Genes/genes.gtf <SampleOutputFolder1>/
accepted_hits.bam> <SampleOutputFolder2>/accepted_hits.bam
```

Parameter	Description
`<Sample list>`	Comma-delimited list of labels for the samples.
`<OutputDir>`	Location to store output files.
`<iGenomesFolder>`	Path and folder where the genome is stored.
`<SampleOutputFolder1>`, `<SampleOutputFolder2>`	Paths and folders where the two SAM alignment files are stored.

Example

This example uses the test data to compare the TruSeq Ribo Profile and total RNA samples.

NOTE
Perform this analysis using replicates on each condition, if possible.

```
> cuffdiff -L SEC,cushion,total -o cuffdiffOutput iGenomes/Homo_
sapiens/UCSC/hg19/Annotation/Genes/genes.gtf ./TopHatOut/Sample_
```

```
494_5000/accepted_hits.bam ./TopHatOut/Sample_494_10000/accepted_
hits.bam ./TopHatOut/Sample_494_11000/accepted_hits.bam
```

This command will create several files in the specified output directory. The expression of both genes and specific isoforms are directly compared in the files gene_exp.diff and isoform_exp.diff. For a discussion of other files generated by this command, see the Cufflinks manual at http://cufflinks.cbcb.umd.edu/manual.html.

Visualizing Results in IGV

Integrative Genomics Viewer (IGV; http://www.broadinstitute.org/igv/) can be used to visualize the RNA-Seq results. This part of the workflow describes how to visualize the alignments, junction counts, and gene models. The files coverage.wig and junctions.bed can be used for immediate visualization of the TopHat alignments as custom tracks in IGV.

IGV Input Files

IGV uses the following input files:

- Junction counts (junctions.bed)
- Gene models in GTF format
- Alignment files in BAM format

Some of the TopHat output files must be modified for better visualization; this process is described in the next section.

Indexing BAM Files

BAM files must be indexed for viewing in IGV:

```
>samtools index <SampleInputSortedBAM>
```

Parameter	Description
<SampleInputBAM>	Sorted BAM file to be indexed (with .bam extension).

NOTE
SAM/BAM file visualization is useful only at the read alignment level. When zooming out, no coverage is visible. To view overall coverage across a chromosome, upload coverage.wig. Starting with TopHat v1.1, there is no coverage.wig file produced. Use IGVtools or BEDTools to make the coverage track from the BAM file.

Example

The following example uses the commands for the human sample test data:

```
> cd ./TopHatOut/Sample_494_5
> samtools index accepted_hits.bam
> cd ..
```

Junction Counts

In each alignment directory, the junctions.bed file contains the number of reads aligned to each intron junction. It is possible to load this file into IGV immediately. However, Illumina recommends first making a few modifications, including adding a track name and replacing the TopHat identifier with the number of supporting reads. This process is accomplished with the following commands:

```
>echo "track name=<SampleID> description=\"<SampleID>\"" >
<SampleOutputFolder>/modified_junctions.bed
>tail -n +2 <SampleOutputFolder>/junctions.bed | perl -lane 'print
join("\t",(@F[0..2,4],$F[4]*50,@F[5..$#F]))' >>
<SampleOutputFolder>/modified_junctions.bed
```

Parameter	Description
`<SampleID>`	Prefix for the output files.
`<SampleOutputFolder>`	Path and folder where the output for the samples under comparison is stored.

Example

The following example uses the commands for the human sample test data:

```
>echo "track name=Sample_5 Junctions description=\" Sample_5
Junctions\"" > Sample5_output/modified_junctions.bed
>tail -n +2 Sample5_output/junctions.bed | perl -lane 'print join
("\t",(@F[0..2,4],$F[4]*50,@F[5..$#F]))' >> Sample5_
output/modified_junctions.bed
```

Final IGV Visualization

Load each of the output files generated with the IGV browser. If you are copying the result files to an alternate computer for visualization, you must copy the alignment index (`<SampleID>.bam.bai` files), as well as the alignment itself (`<SampleID>.bam`). Loading these values will produce the visualization, allowing you to see coverage, aligned reads, and junction counts for each of the 2 tissues. Consider also uploading the GTF file for the transcripts model, along with a junction track that shows the number of reads supporting each junction and a comparison to a known annotation track (eg, the Refseq transcripts in Broad IGV). This process will help compare your data to known transcripts and see differences between different samples.

NOTE
For large genome regions, Illumina recommends uploading separate coverage tracks, because BAM coverage is shown only for smaller regions. Loading a full BAM file for large regions with highly expressed transcripts may kill or freeze the browser.

TruSeq Ribo Profile Quality Metrics

Length Distribution

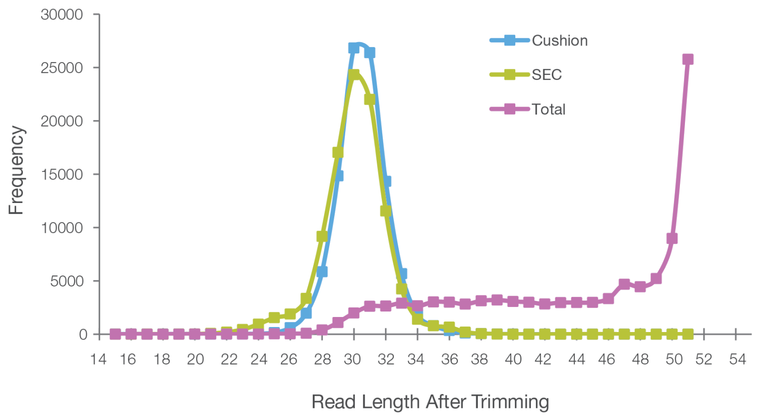
After contaminant removal and alignment with TopHat, most of the reads have the expected distribution of ribosome footprint lengths in the TruSeq Ribo Profile sample. Footprint lengths normally peak around 28–31 bases, depending on species, experimental condition, and alignment parameters.

In order to get an idea for the length distribution of the aligned reads, use the following command on the TopHat output:

```
>samtools view accepted_hits.bam | grep -E '(NM:i:0)|(^@)'| awk -v
v1=0.05 'BEGIN {srand()} !/^$/ { if (rand() <= v1) print length
($10)}' | head -n 100000 | sort | uniq -c >
alignedLengthHistogram.txt
```

The output file, alignedLengthHistogram.txt will give you an approximate length distribution of the 100,000 randomly selected reads with perfect alignments. If you plot the second column versus the first column of the output file, you will get an estimate of the read-length distribution. Figure 4 shows a read-length distribution for footprinted samples (SEC and sucrose cushion) and for the total RNA sample.

Figure 4 Distribution of Trimmed Read Lengths



The mean values for the total RNA sample (total), sucrose cushion sample (cushion), and SEC sample (SEC) are 44, 30, and 31, respectively.

Samples generated with ribosome profiling should also show an increased proportion of reads aligning to mRNA and especially within the coding sequence of annotated transcripts. Use the following command with Picard Tools 4 to visualize the relative distribution of aligned reads on annotated gene models. To install the toolbox, see http://picard.sourceforge.net.

```
> java -jar <Path to Picard>/CollectRnaSeqMetrics.jar REF_
FLAT=<Path t iGenomes>/Annotation/Genes/refFlat.txt.gz STRAND_
SPECIFICITY=NONE CHART_OUTPUT=<output file name>".pdf" INPUT=<path
to TopHat output/accepted_hits.bam>
OUTPUT=<sampleID>".metrics.txt"
```

Parameter	Description
`<Path to iGenomes>`	Path and folder of the iGenomes directory.
`<sampleID>`	Name of the sample being analyzed.
`<path to TopHat output>`	Path to the TopHat output containing the accepted_hits.bam file.

For further information on the inputs and outputs of the CollectRnaSeqMetrics toolbox, see:
http://picard.sourceforge.net/command-line-overview.shtml#CollectRnaSeqMetrics

Example

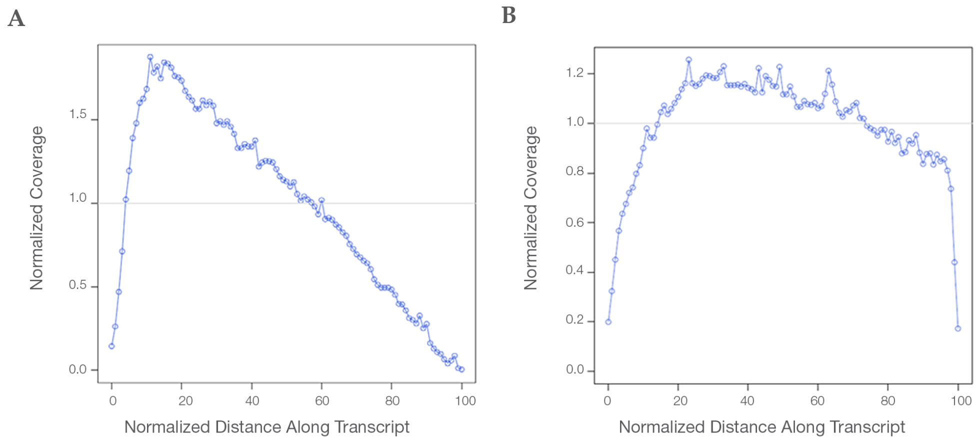
The following example uses the commands for the human sample test data:

```
> java -jar picard/CollectRnaSeqMetrics.jar REF_
FLAT=iGenomes/HomoSapiens/UCSC/hg19/Annotation/Genes/refFlat.txt
.gz STRAND_SPECIFICITY=NONE CHART_OUTPUT=Sample_5"metrics.pdf"
INPUT==./TopHatOut/Sample_494_5/accepted_hits.bam OUTPUT=Sample_
5".metrics.txt"
```

The commands result in 2 files:

1. *.metrics.txt file. This file contains collect metrics about the alignment of RNA to various functional classes of loci in the genome: coding, intronic, UTR, intergenic, ribosomal. The output columns PCT_CODING_BASES, PCT_UTR_BASES, PCT_INTRONIC_BASE, PCT_INTERGENIC_BASES can be plotted as shown in Figure 5.
2. *.metrics.pdf file. This PDF file is a graphic with the average read density (Figure 5) across the length of the 1,000 most highly expressed transcripts. Detailed calculations are provided in the Picard Tools documentation.

Figure 5 Output from Picard Tools



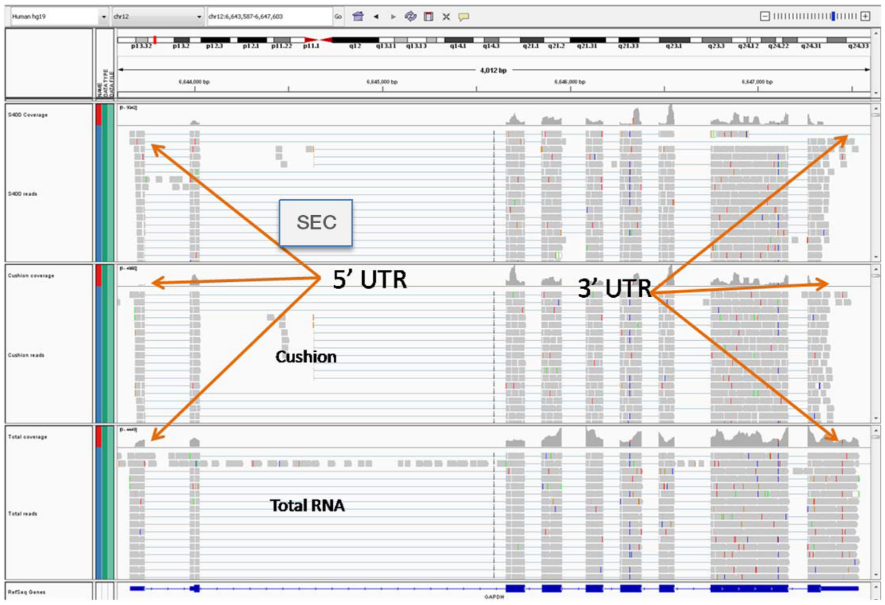
RNA-Seq coverage vs. transcript position for all reads in the accepted_hits.bam file. The CollectRNASeqMetrics method shows the 5′ to 3′ coverage across the 1,000 most expressed transcripts.

A RPF, SEC sample
B Total RNA sample

For TruSeq Ribo Profile samples, higher read densities occur near the 5′ end of transcripts, compared to standard mRNA-Seq. The addition of cyclohexamide in the TruSeq Ribo Profile proecdure halts translation elongation, resulting in a higher density of ribosomes near the 5′ end of transcripts than the 3′ end.

Another distinctive feature of TruSeq Ribo Profile samples is the depletion of read density in untranslated regions (UTRs) and, often, an abrupt spike in density at the start and stop codon. Figure 6 shows the read density profiles at the 5′ UTR (A) and 3′ UTR for the *GAPDH* gene for total RNA versus RPF samples. In the regions indicated, the read density profile extends over the entire UTR regions for the total RNA regions, whereas reads are markedly absent from the RPF samples.

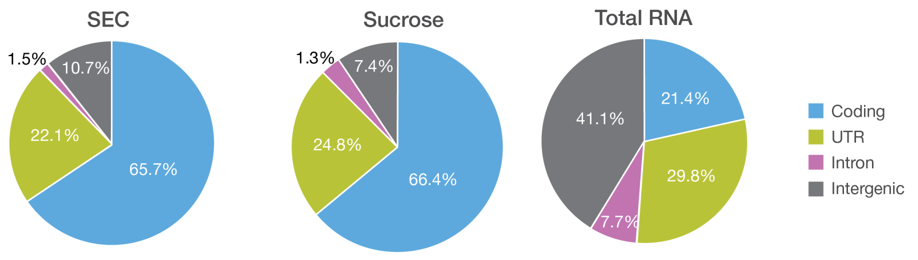
Figure 6 IGV Screenshot



The example shows the GAPDH region for footprinted samples (SEC, sucrose cushion) and the total RNA sample.

A plot of the distribution of reads in various genomic features (Figure 7) also shows differential read densities for SEC, sucrose cushion, and total RNA samples. TruSeq Ribo Profile samples (panels A and B) contain a much larger fraction of reads aligning to coding regions and a greatly reduced percentage of reads in intronic and intergenic regions.

Figure 7 Read Density Distribution

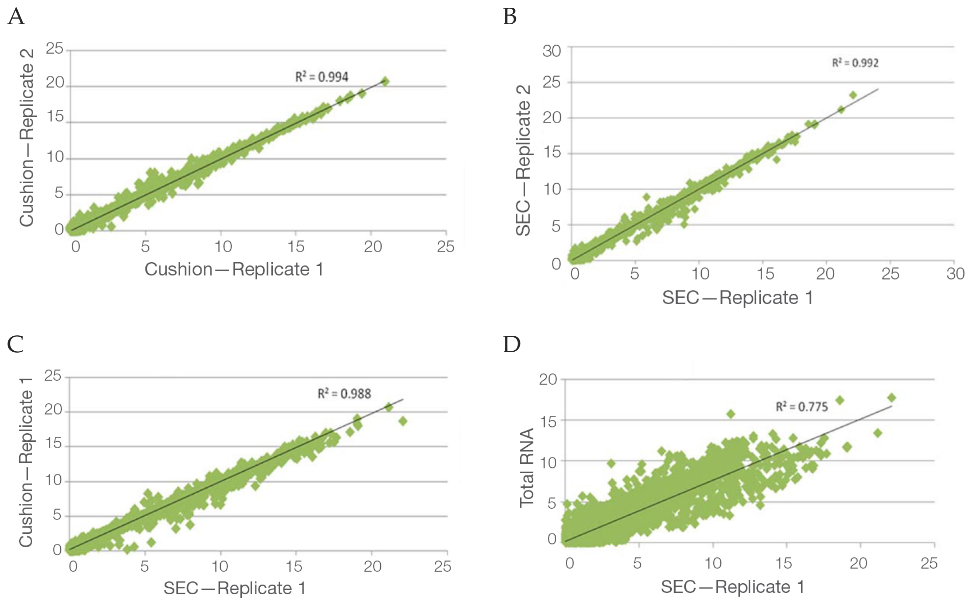


Read density distribution on various annotation features output by Picard tools.

A SEC RPF sample
B Sucrose cushion RPF sample
C Total RNA sample

The TruSeq Ribo Profile protocol is also highly reproducible between replicates. In Figure 8, panels A and B show highly reproducible gene counts ($\log_2$ FPKM) between 2 technical replicates of sucrose cushion and SEC samples, respectively, with Pearson correlation coefficients of > 0.99. Further, panel C shows highly reproducible expression between the same library processed uisng a sucrose cushion and SEC. Panel D demonstrates the correlation between expression levels in total RNA and SEC samples. As expected, slightly lower gene correlations are observed than for RPF to RPF comparisons.

Figure 8 Sample Reproducibility



Technical replicates of TruSeq Ribo Profile samples show high reproducibility between replicates of cushion and SEC samples (A and B) as well as between cushion and SEC samples (C). SEC and total RNA samples are compared in (D). Axes are $\log_2$(FPKM); genes with counts less than 1 FPKM were not included.

References

1 Trapnell C, Pachter L, Salzberg SL. TopHat: discovering splice junctions with RNA-Seq. *Bioinformatics*. 2009;25(9):1105-1111.

2 Langmead B, Trapnell C, Pop M, Salzberg SL. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol.* 2009;10(3):R25.

3 Trapnell C, Roberts A, Goff L, et al. Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks. *Nat Protoc.* 2012;7(3):562-578.

Appendix

Software Installation

Bowtie

Download the Bowtie executables from http://sourceforge.net/projects/bowtie-bio/files/bowtie/ and unzip the file.

```
> unzip bowtie-0.12.7-linux-x86_64.zip
```

Add the bowtie and bowtie-build executables in a separate bin directory where all the executables are located or add the location of the folder to your path, eg:

```
> export PATH=<mylocation>:$PATH
```

TopHat

Download the binary package (TopHat-1.4.1.Linux_x86_64.tar.gz) for version 1.4.1 of TopHat from http://TopHat.cbcb.umd.edu/downloads/. Unpack the tar file using the following command:

```
> tar -zxvf TopHat-1.4.1.Linux_x86_64.tar.gz\
> cd TopHat-1.4.1.Linux_x86_64
```

Add the contents of the folder in a separate bin directory where all the executables are located, or add the location of the TopHat folder to your path.

Cufflinks

Download the binary package of version 1.3.0 for Cufflinks (http://cufflinks.cbcb.umd.edu/downloads/) and unpack the Cufflinks tarball.

```
> tar -zxvf cufflinks-1.3.0.Linux_x86_64.tar.gz
> cd cufflinks-1.3.0.Linux_x86_64
```

Add the contents of the folder in a separate bin directory where all the executables are located, or add the location of the TopHat folder to your path.

Samtools

Download the executable from http://sourceforge.net/projects/samtools/files/samtools/ and unpack the samtools tarball. Change directory to the samtools folder and create the executable with `make`:

```
> tar -jxvf samtools-0.1.18.tar.bz2
> cd samtools-0.1.18
> make
```

Add the samtools executable in a separate bin directory where all the executables are located or add the location of the samtools directory to your path.

FASTX toolkit

Download precompiled binaries from: http://hannonlab.cshl.edu/fastx_toolkit/download.html. Click the **Download precompiled binaries** section, select the appropriate executable, for example **Linux (64bit)**. The command for unpacking the tarball will write the executables to a bin folder. Therefore, create a fastx folder prior to unpacking.

```
> mkdir fastx_toolkit_0.0.13.2
> mv fastx_toolkit_0.0.13_binaries_Linux_2.6_amd64.tar.bz2
```

```
   > cd fastx_toolkit_0.0.13.2
   > tar -jxvf fastx_toolkit_0.0.13_binaries_Linux_2.6_amd64.tar.bz2
```

The fastx executables will be located in the folder fastx_toolkit_0.0.13.2/bin. Either add the executable to an existing bin directory where the other executables are located, or add the location of the fastx_toolkit_0.0.13.2/bin directory to your path, eg:

```
   >export PATH=<my fastx_toolkit_0.0.13.2 location>:$PATH
```