ARTseq™ Bioinformatics User Guide rev.0.1

INTRODUCTION

This guide provides suggestions for data analysis of libraries prepared using Epicentre's ARTseq™ Ribosome Profiling Kits, and sequenced on an Illumina sequencer.

IMPORTANT NOTICE: This document provides information on data analysis for ARTseq™ Ribosome Profiling libraries that has been demonstrated internally and may be of interest to customers. The information is provided as-is and is not an Epicentre product and is not accompanied by any rights or warranties. Customers using or adapting this information should obtain any licenses required and materials from authorized vendors. Epicentre products mentioned herein are for research use only unless marked otherwise. While customer feedback is welcomed, this user guide is not supported by Epicentre Technical Support and Field Application scientists.

There are 4 main steps in the ARTseq analysis pipeline:

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

Following alignment, the resulting BAM file can be used for downstream analyses with other bioinformatics tools. In this document, we will demonstrate the use of the Cufflinks package[1] for transcript annotation, assembly, quantification and visualization.

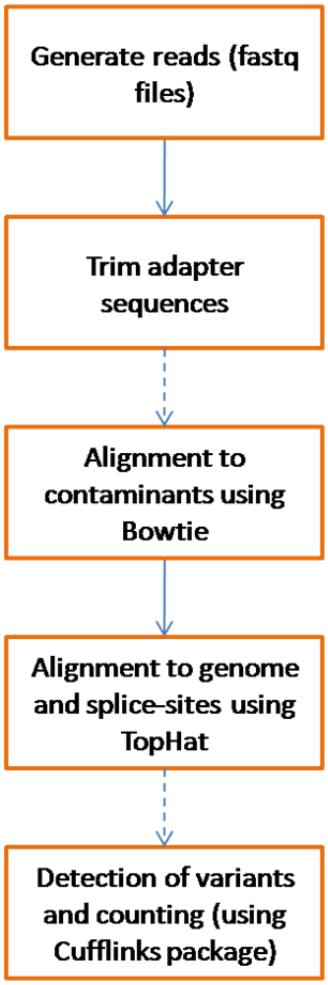

Figure 1. Overview of the ARTSeq analysis pipeline. Optional steps are indicated in dashed lines.

While this document provides general instructions on running analysis, it is not intended to be a comprehensive guide. Please also note that TopHat is not an Illumina/Epicentre supported product, but an open source initiative.

The protocol described in this document also assumes you will have access to a UNIX server with command-line access and 4GB of RAM (*64-bit machine with 16GB RAM preferred*) and basic knowledge of UNIX. In addition, you may need additional assistance from your institute's IT department for software installation and permissions.

Commands on a UNIX environment are prefixed with a ">" symbol and displayed in a `modified font`. Arguments and parameters to be provided by the user are shown in angled brackets and in bold font .

Software Installation

This workflow requires installation of the following tools:

- CASAVA1.8 (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 0.0.13.2 (http://hannonlab.cshl.edu/fastx_toolkit/index.html)

For further installation instructions, please refer to Appendix I at the end of this document or the "Getting started" or "Download' link on each webpage referenced above. The following versions of these packages were used in the development of this workflow: Bowtie(0.12.7), TopHat (v1.4.1), Cufflinks (v1.3.0), Samtools (0.1.18) and FASTX toolkit 0.0.13.2, though more recent versions of the programs will also be compatible with this pipeline. In addition to these tools, this workflow also assumes the availability of the following command-line tools and Perl: gunzip, wget, awk.

Once the tools have been installed, you will need to ensure 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, please enter: (assuming bash shell).

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

Testing the Installation

To verify that all the tools are properly installed and can be run on the command line, please execute the following commands:

- Bowtie:
  ```
  >bowtie --version
  ```

  This should return something 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 should return something similar to the following:
  TopHat v1.4.1
  …

- Cufflinks:

  ```
  >cufflinks
  ```

  This should return something similar to the following:
  Cufflinks v1.3.0
  …

- FASTX toolbox

  ```
  >fastx_trimmer -h
  ```

  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.
  ...

Please verify that there are no errors. If any of these commands generate an error, please check your installation.

Genome Reference Files

In order to align ARTSeq samples to common contaminants and a reference genome, you will need to download genome annotation and reference files. 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 Illumina's FTP site : ftp://ussd-ftp.illumina.com

For example, you can download the genome sequences, annotation, and bowtie index files 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.
```

You can login using the following credentials:
- Username: igenome
- Password: G3nom3s4u

The downloaded file then needs to be unpacked using the following command:

```
> tar xvzf Homo_sapiens_UCSC_hg19.tar.gz
```

Unpacking will make its own folder
Homo_sapiens/UCSC/hg19

Within this folder, the following subfolders will be present:
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

We are providing three samples along with this guide. There is also a README file which indicates what these samples are and a list of commands from this guide all in one file.
URLS listed below:
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:

1- Artseq sample
2- Sucrose gradient sample
3- Total RNA sample

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>
```

Where:

• <**WorkflowFolder**> is the path and folder where the results should be stored

Step 1: 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 these in separate directories. Each directory can be independently analyzed 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 one 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 as described above.

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 needs a BaseCalls directory and a sample sheet to start a run. These files are described below (fig.2).

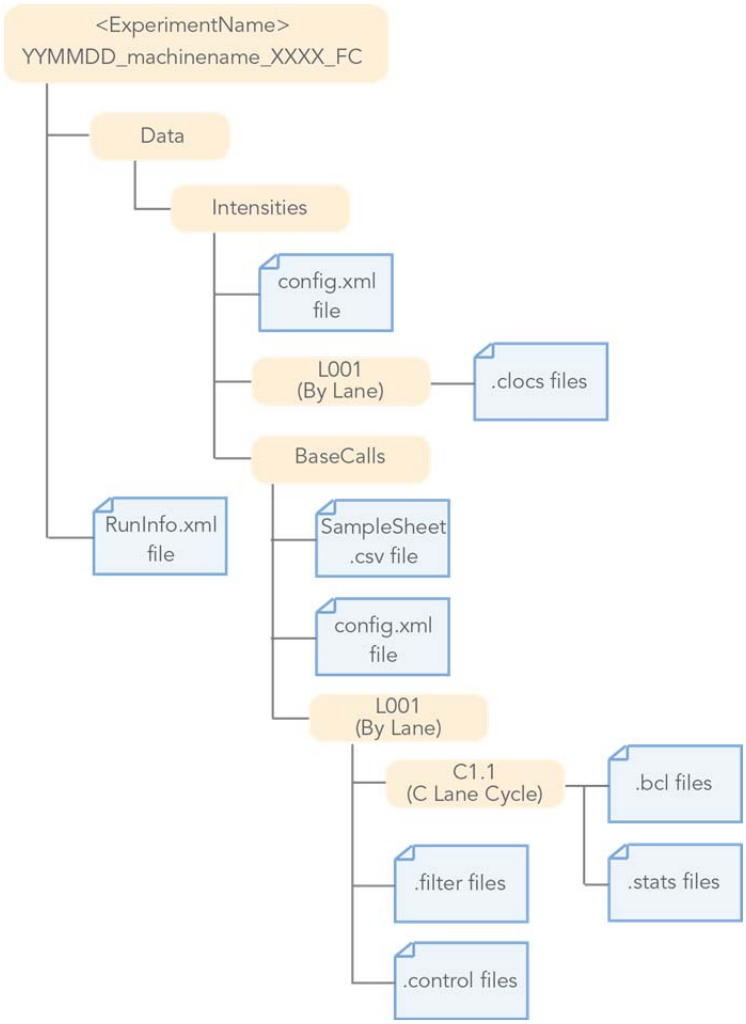

Figure 2. Folder structure required for the CASAVA demultiplex program.

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 it looks for based on the RTA version that was used to generate them.
- RunInfo.xml file. The RunInfo.xml is 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.

The sample sheet contains the following 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 (for example ACCAGTAA-GGACATGA)
Description	Description of the sample
Control	Y indicates this lane is a control lane, N means samples
Recipe	Recipe used during sequencing
Operator	Name or ID of the operator
SampleProject	The project the sample belongs to

You can generate it using Excel or other text editing tool that allows .csv files to be saved. Enter the columns specified above 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.

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 Index

As of CASAVA 1.8, you can assign samples without 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 one script, configureBclToFastq.pl. This section describes how to perform bcl conversion and demultiplexing in CASAVA 1.8.

Usage of configureBclToFastq.pl

The standard way to run bcl conversion and demultiplexing is to first create the necessary Makefiles, which configure the run. Then you 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. We suggest 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 done, review the analysis for each sample.

Options for Bcl Conversion and Demultiplexing

The options for demultiplexing are described below.

--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, --	Path to demultiplexed output. Defaults to	--output-dir

output-dir	<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.	<run_folder>/Unaligned
--positions-dir	Path to a directory containing positions files. Defaults depends on the RTA version that isdetected.	--positions-dir<positions_dir>
--positions-format	Format of the input cluster positionsinformation. 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	--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 secondread (Y50) • Ignore next (n)

--ignore-missing-stats	Fill in with zeros when *.stats files are missing	--ignore-missingstats
--ignore-missing-bcl	Interpret missing *.bcl files as no call	--ignore-missing-bcl
--ignore-missingcontrol	Interpret missing control files as not-set control bits	--ignore-missingcontrol
--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 two adapters 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	<adapterdir>/adapter.fa
--man	Print a manual page for this command	--man
-h, --help	Produce help message and exit	-h

Bcl Conversion Output Folder

The directory structure which results from BCL conversion output is shown below 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 the 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 samplesheet. It may be that you need to set the --use-bases-mask option to the length of the index in the sample sheet + the character 'n' to account for phasing. Note that you will not be able to see which indices have been placed in the 'Undetermined_indices' folder

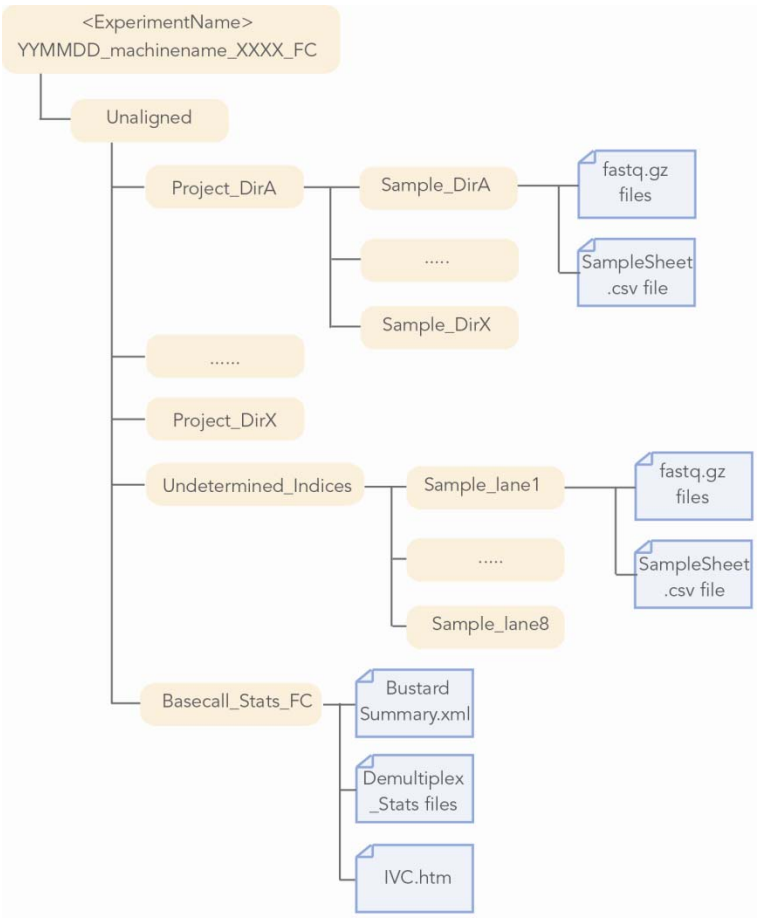

Figure 3. Folder structure required for the CASAVA demultiplex program.

Step 2: Read pre-processing and adapter removal with FASTX toolbox

Due to the short length of ribosome protected fragments (RPF) relative to read lengths, we expect that most reads from ARTSeq libraries will contain 3' adapter sequences. For compatibility with the Bowtie and TopHat aligners, the adapter sequences need to be trimmed off from the 3' end of the read. We recommend the use of the FastX-toolkit. The tool can be downloaded from the following page : http://hannonlab.cshl.edu/fastx_toolkit/download.html

Further instructions on installing the FastX toolbox can be found in Appendix I.

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

```
> 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
```

Where :

- **<Path To Fastq>** is the location of the fastq files generated during bcl conversion including the Project and Sample names e.g. <my run folder>/Unaligned/Project_myProject/Sample6
- **<adapter sequence>** is the adapter sequence used in library preparation. For ARTSeq this sequence is AGATCGGAAGAGCACACGTCT
- **<sampleID>** is the name of the sample being converted
- **<length>** is the minimum length of read to maintain after trimming the adapter sequence. The default value is 5, but we recommend a value of 20-25 depending on the target footprint length
- **<cycle to begin alignment>** is the number of the first cycle to consider for alignment which are adjusted to account for any 5' specific errors like non-templated 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 are available on the Hannon lab website :

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 ./Trimmed**/<sampleID>** folder

Test Data

The above command for the human sample test data, sample would be

```
>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 above produced the file ./Trimmed/Sample_494_5/trimmed.fastq.

The output from the trimmer program removed 9.5% of the input reads and resulted in the output below.

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.

Step 3: Contaminant Removal

In order to remove and quantify the 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. Please note that these steps can be skipped at this stage as well and contaminating sequences masked out post-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 for contaminant alignment

A list of common contaminants including ribosomal and mitochondrial sequences is available via the iGenomes server for selected organisms listed in the iGenomes directory. For example, the ribosomal 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 following files: RDN18-1.fa, RDN25-1.fa, RDN37-1.fa, RDN58-1.fa, which are 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, you will need to copy your sequences and paste your sequences in fasta format. For example,

>gi|555853|gb|U13369.1|HSU13369 Human ribosomal DNA complete repeating unit
GCTGACACGCTGTCCTCTGGCGACCTGTCGTCGGAGAGGTTGGGCCTCCGGATGCGCGCGGGGCTCTGGC
CTCACGGTGACCGGCTAGCCGGCCGCGCTCCTGCCTTGAGCCGCCTGCCGCGGCCCGCGGGCCTGCTGTT
.....

To make the contaminant sequences compatible with Bowtie, we need to 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 as follows:

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

Where <indexName> is the name given to the ebwt files

In the example below, we are creating a directory containing ebwt ribosomal sequences in a 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 in the previous step.

To account for the short read length values generated by RPF fragments, 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, please go to 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>** as specified in the command below:

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

Where

- <path to ebwt rRNA folder> is the folder generated with ebwt files for contaminants
- <indexName> is the name of the ebwt files
- <trimmed fastq file> points to the fastq file generated with clipped sequences
- <outfile name> is the name assigned to the file to contain the rRNA sequence alignments.

For example, following our analysis for Sample_5 of our test data, we would 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 summary of the alignment in terms of the number of input sequence and the number of reads aligned to the rRNA sequences will be contained in the file stats.txt.

Removal of additional contaminant sequences

In addition to rRNA, your sample could have other contaminating sequences such as 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 result in cleaving the individual molecules in half. This results in two fragments that are roughly similar in size to RPF 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 could remain in the same.

For tRNA alignments, you can simply combine tRNA sequences with the rRNA sequences from the step above and run one single contaminant removal analysis. Alternately, you can choose to have another iteration of bowtie alignment against tRNA sequences if you would like to quantify the tRNA contamination separately from rRNA.

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

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

2. Create a Bowtie index of your file (ebwt files) e.g.

```
> 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 your contaminant and the input sequence reads. For example, if we wish to align the reads after rRNA removal to tRNA sequences, we could run the following alignment:

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

Where :

- `< unalignedSequences >` is the name of the file the unaligned sequences will go to. In this example, all reads which do not map to rRNA and tRNA will be in this file.
- `< input fastq file>` is the name of the input fastq file
- `<indexName>` is the name of the ebwt files
- `<aligned sequences>` is the name of the file the alignments will be written to

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

For example, following our analysis for Sample_5 of our test data, we would enter the following command:

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

Step 4: Alignment to genome and splice junctions

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
```

Where:

- <SampleOutputFolder> is the path and folder where the sample output will be stored
- <iGenomesFolder> is the path and folder of the iGenomes directory
- <input fastq> is the name of the fastq to submit for alignment to the genome. This file is the same one that was generate after the previous contaminant removal step.

The command above tells TopHat to align based on the gene model defined in the GTF file provided.

NOTE:
We also suggest 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.

Test Data

For example, for the test data this would be the command to run on a multi-core machine:

```
>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 BAM-format 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 and is compatible with many other third party bioinformatics tools for analysis.

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

Step 5 (Optional): Detecting Transcripts and Counting

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 has three separate tools (Cufflinks, Cuffmerge, and Cuffdiff) for the following tasks:

- 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.

In this document we will focus only on gene/transcript quantification using a known reference genome and differential expression across samples. For further instructions on the Cufflinks suite for tasks such as novel junction discovery please review the Trapnell et al publication (reference 5) in Nature Protocols.

Running Cufflinks for Quantifying Genes and Transcripts

If CuffLinks is provided an 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. The following command can be used 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 ..
```

Where:
• <SampleOutputFolder> is the path and folder where the TopHat output is located.
• <iGenomesFolder> is the path and folder where the genome is stored

WARNING
If you run Cufflinks in the directory where there is already output from another cufflinks it will be over-written. If you want to keep different cufflinks outputs run each cufflinks from different directory as it always prints output in 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.

Test Data

For example, for the test data, this would be the following:

```
>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 (ref 5).

Quantification Output

In each output directory, we will have created two 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 terms of FPKM, or Fragments Per Kilobase of sequence per Million mapped reads. In simple terms, this measure normalizes the number of aligned reads by the size of the sequence feature and the total number of mapped reads

Cuffdiff - Differential Expression

Cuffdiff will perform a differential expression analysis between two samples on

annotated genes. Run Cuffdiff, providing a GTF file and the two SAM-format alignment files, the following way:

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

Where:

- <SampleList> is a comma-delimited list of labels for the samples<OutputDir> is the location to store output files
- <iGenomesFolder> is the path and folder where the genome is stored
- <SampleOutputFolder1> and <SampleOutputFolder2> are the paths and folders where the two SAM-format alignment files are stored

Test Data

For example, for the test data, we can compare the ARTSeq and total RNA samples. Note: it is preferred to do 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, please see the 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 results of the RNA sequencing results. This part of the workflow will describe how to visualize the alignments, junction counts, and gene models. ***coverage.wig*** and ***junctions.bed*** can be used for immediate visualization of the TopHat alignments as custom tracks in the 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 should be slightly modified for better visualization; this

is described below.

Indexing BAM Files

The BAM files must be indexed to view in IGV:

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

Where:

- <SampleInputBAM> is the sorted BAM file that needs to be indexed (with .bam extension)

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

Test Data

For the test data, indexing would be the following:

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

Junction Counts

In each alignment directory, the file ***junctions.bed*** contains the number of reads aligned to each intron junction. While it is possible to immediately load this file into IGV, we will first make a few useful modifications, including adding a track name and replacing the TopHat identifier with the number of supporting reads. This is accomplished with the following command:

```
>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
```

Where:

- <SampleID> is the prefix for the output
- <SampleOutputFolder> is the path and folder where the output is stored of the samples being compared

Test Data

For example, for the test data, this would be the following:

```
>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
```

IGV Visualization

Load each of the output files generated above with the IGV browser (http://www.broadinstitute.org/igv/). Note that 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 produced the following visualization, allowing you to see coverage, aligned reads, and junction counts for each of the two tissues. You may also want to upload 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 (for example, the Refseq transcripts in Broad IGV). This will help you compare your data to known transcripts and see differences between different samples.

NOTE

For large genome regions we recommend uploading separate coverage track as BAM coverage is shown only for smaller regions. Also please aware that loading a full BAM file for large regions with very highly expressed transcript may kill or freeze the browser.

ARTSeq QC Metrics

Length Distribution

After contaminant removal and alignment with TopHat, we should observe that most of our reads have the expected distribution of ribosome footprint lengths in our 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, you can 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 distribution of the read lengths. Figure 4 below shows a read length distribution for footprinted samples (SEC and cushion) and for total RNA samples.

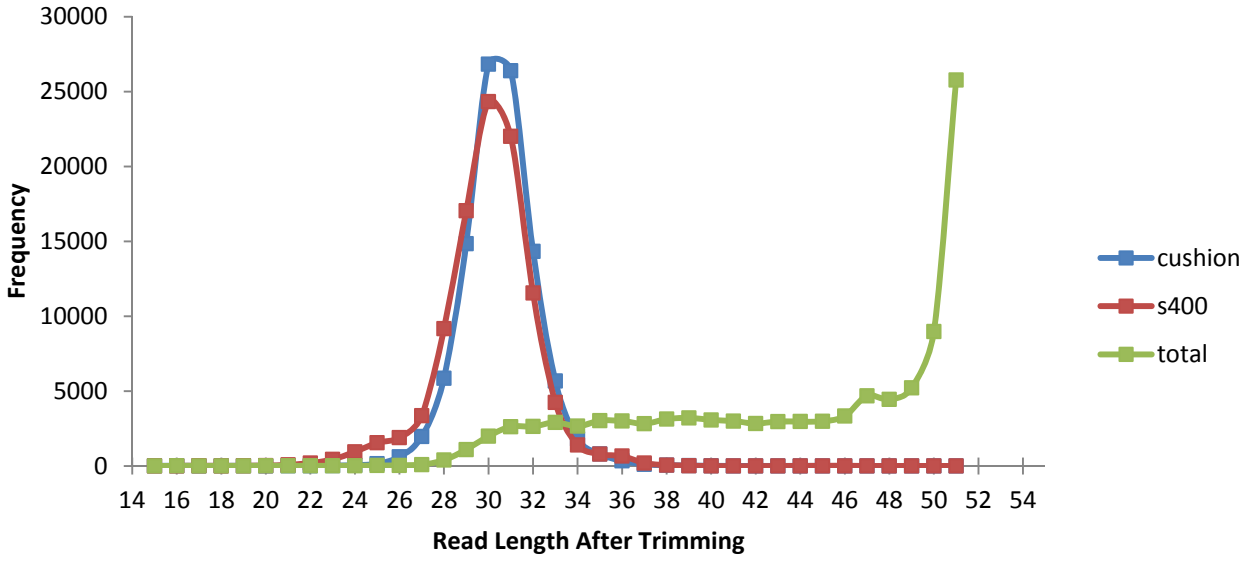

Figure 4. Distribution of trimmed read lengths. The mean for the total RNA sample, cushion sample and spin column samples are 44,30 and 31 respectively.

Samples generated with ribosome profiling should also show an increased proportion of reads aligning to mRNA and especially within the CDS of annotated transcripts. The following command using Picard tools [4] can be used to visualize the relative distribution of aligned reads on annotated gene models. To install the toolbox, please refer to http://picard.sourceforge.net/.

```
> java -jar <Path to Picard>/CollectRnaSeqMetrics.jar REF_FLAT=<Path
to 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"
```

where:

- <**Path to Igenomes**> is the path and folder of the iGenomes directory
- <**sampleID**> is the name of the sample to analyze
- <**path to TopHat output**> is the path to the TopHat output containing the 'accepted_hits.bam' file

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

For our human sample, the command would be:

```
> 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 results in 2 files:

- *.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 8 below..

1. *.metrics.pdf file. This file is a pdf file with a graphic with the average read density (fig.5) across the length of 1000-most highly expressed transcripts. For details of the calculations please refer to the Picard Tools link above.

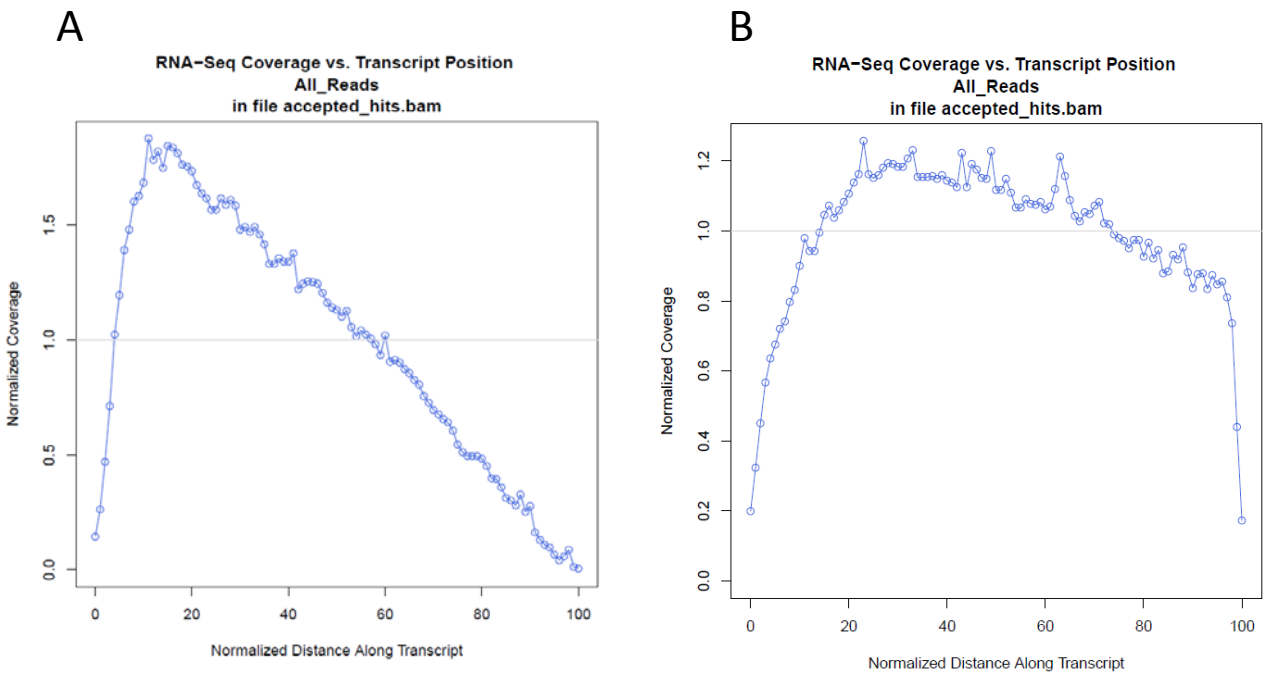

Figure 5. Output from Picard Tools *CollectRNASeqMetrics* method which shows the 5' to 3' coverage across the 1000-most expressed transcripts. for A) RPF, SEC sample and B) total RNA sample.

For ARTSeq samples, we will see higher read densities near the 5' end of the transcripts compared to standard mRNA seq. Upon the addition of cyclohexamide, translation elongation is halted resulting in a higher density of ribosome closer to the 5’ end of transcripts than the 3’ end.

Another distinctive feature of ARTSeq samples is the depletion of read density in UTR regions and often an abrupt spike in density at the start and stop codon. Figure 6 below 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, we observe that the read density profile extends over the entire UTR regions for the total RNA regions whereas reads are markedly absent from the RPF samples.

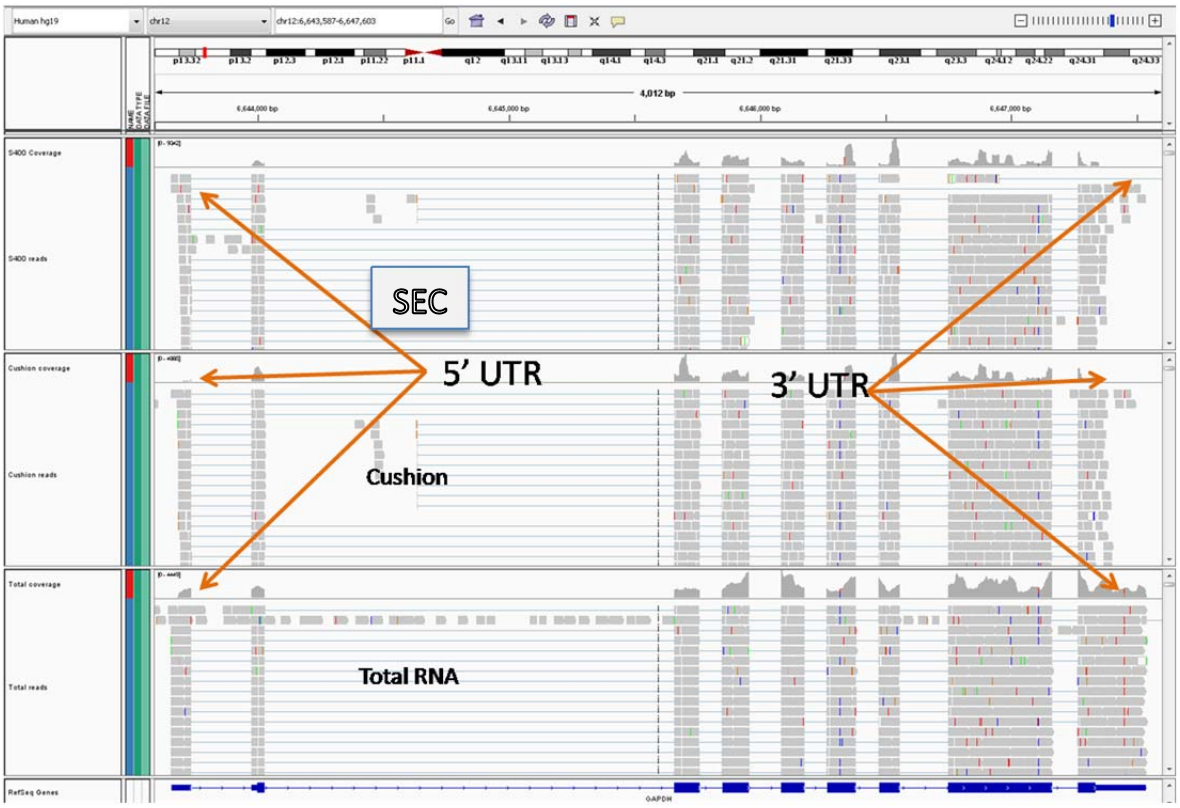

Figure 6. IGV screenshot of the GAPDH region for footprinted samples (SEC, cushion) and a total RNA sample.

A plot of the distribution of reads in various genomic features (figures 7 A B,C below) also shows differential read densities for SEC, cushion and total RNA samples. From this figure, we see that ARTSeq samples (fig 7A and 7B) contain a much larger fraction of reads aligning to coding regions and a greatly reduced percentage of reads in intronic and intergenic regions.

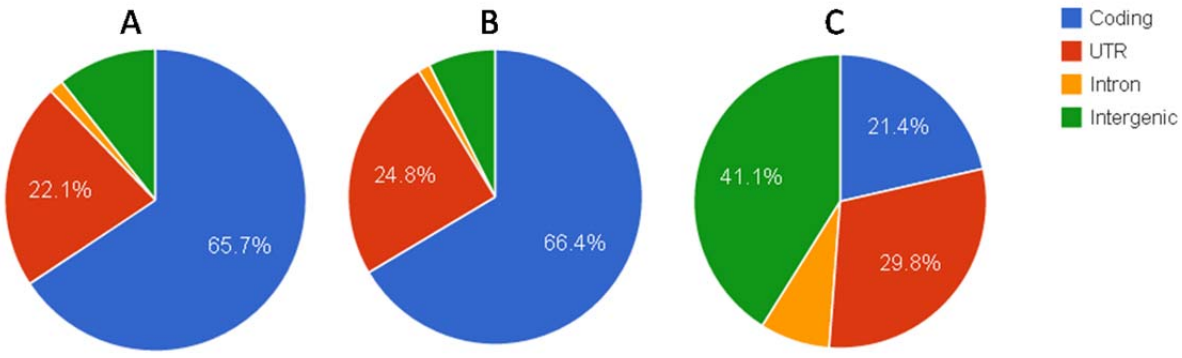

Figure 7. Read density distribution on various annotation features output by Picard in A) SEC RPF samples, B) cushion RPF samples and C) total RNA.

The ARTSeq protocol was also shown to be highly reproducible between replicates. Figures 8A and 8B below show highly reproducible gene counts (log2 FPKM) between 2 technical replicates of cushion and spin column samples respectively, with Pearson correlation coefficients of >0.99. Further in figure 8c, we also observe highly reproducible expression between the same library processed on a cushion column and a spin column. Figure 8d demonstrates the correlation between expression levels in total RNA and spin column samples. As expected, we see slightly lower gene correlations compared to RPF to RPF comparsions.

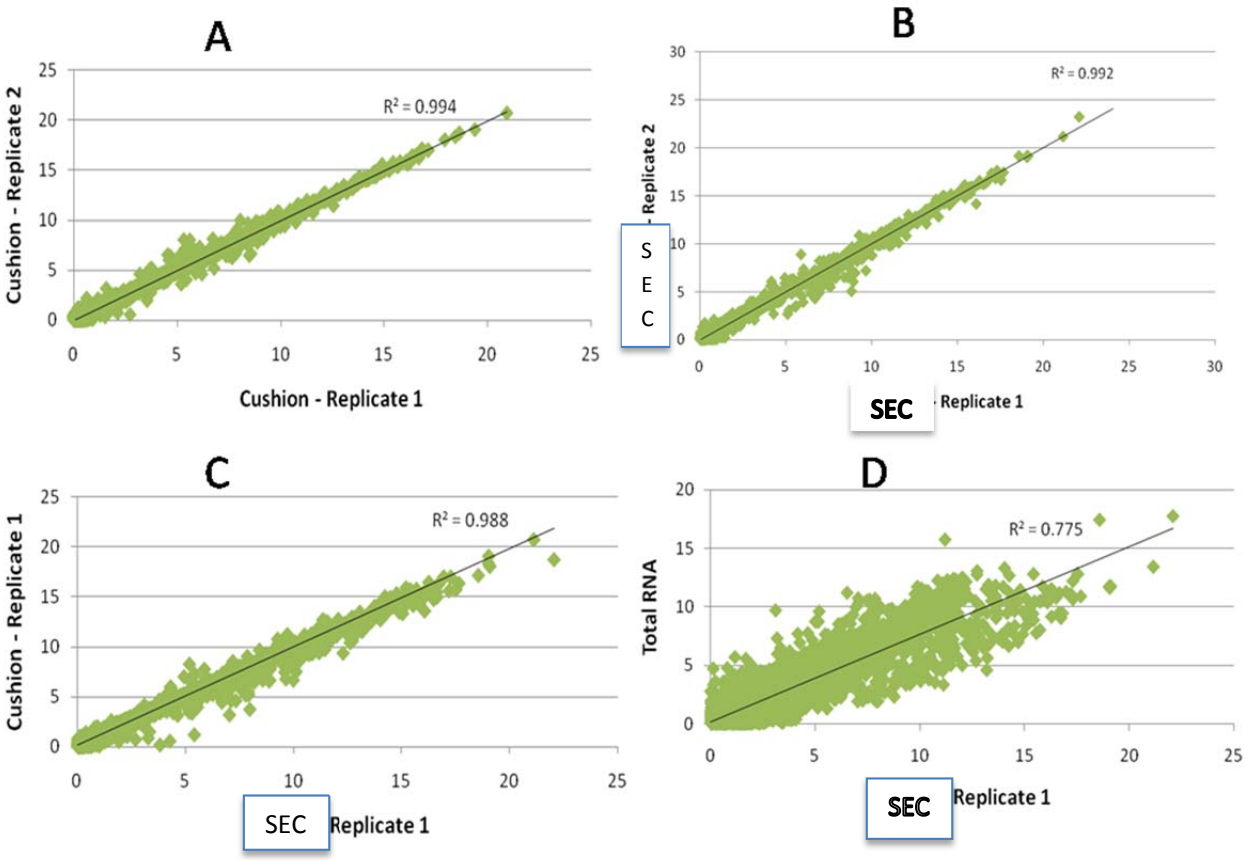

Figure 8. Technical replicates of ARTSeq samples show high reproducibility between replicates of cushion and SEC samples (A and B) as well as between cushion and SEC samples (C). SEC and total RNA samples are compared in (D). Axes are log2(FPKM) and genes with counts less than 1 FPKM were not included.

References

1. Trapnell C, Pachter L, Salzberg SL.(2009) TopHat: discovering splice junctions with RNA-Seq. Bioinformatics 25(9):1105-11.
2. Langmead B, Trapnell C, Pop M, Salzberg SL. (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome Biol 10(3):R25.
3. Ingolia, NT, Brar, GA, Rouskin, S, McGeachy, AM, Weissman, JS. (2012) The ribosome profiling strategy for monitoring translation in vivo by deep sequencing of ribosome-protected mRNA fragments. Nat Prot 7(8): 1534-1550.Trapnell C, Williams BA, Pertea G, Mortazavi A, Kwan G, van Baren MJ, Salzberg SL, Wold BJ, Pachter L. (2010) Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. Nat Biotechnol 28(5):511-5.
4. Trapnell C, Roberts A, Goff L, Pertea G, Kim D, Kelley D, Pimentel H, Salzburg SL, Rinn JL, Pachtor L. (2012). Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks. Nat Prot 7(3): 562-578,

Appendix I - Software Installation

Bowtie

Download the bowtie executables from http://sourceforge.net/projects/bowtie-bio/files/bowtie/, and unzip and 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 e.g.

```
> 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

To install 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

To install the samtools package, download the executable from

(http://sourceforge.net/projects/samtools/files/samtools/) and unpack the SAM tools 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 pre-compiled binaries from http://hannonlab.cshl.edu/fastx_toolkit/download.html. Click the "Download pre-compiled binaries" section, select the appropriate executable, for example "Linux (64bit)". The command for unpacking the tarball will write the executables to a 'bin' folder. As such, we will 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 e.g.

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