Galaxy Training!

1: RNA-Seq reads to counts

Authors: Maria Doyle Belinda Phipson Harriet Dashnow

Overview

Questions:

- How to convert RNA-seq reads into counts?
- How to perform quality control (QC) of RNA-seq reads?
- How to do this analysis efficiently in Galaxy?

Objectives:

- Learn how RNA-seq reads are converted into counts
- Understand QC steps that can be performed on RNA-seq reads
- Generate interactive reports to summarise QC information with MultiQC
- Use the Galaxy Rule-based Uploader to import FASTQs from URLs
- Make use of Galaxy Collections for a tidy analysis
- Create a Galaxy Workflow that converts RNA-seq reads into counts

Requirements:

- Introduction to Galaxy Analyses
- Sequence analysis
  - Quality Control: slides - hands-on
  - Mapping: slides - hands-on
- Transcriptomics
  - Reference-based RNA-Seq data analysis: hands-on
- Using Galaxy and Managing your Data
  - Using dataset collections: hands-on
  - Rule Based Uploader: hands-on

Time estimation: 3 hours

Supporting Materials:

Datasets Workflows FAQs Available on these Galaxies

Last modification: Feb 7, 2023

License: Tutorial Content is licensed under Creative Commons Attribution 4.0 International License The GTN Framework is licensed under MIT

Introduction

Preparing the reads

Introduction

Measuring gene expression on a genome-wide scale has become common practice over the last two decades or so, with microarrays predominantly used pre-2008. With the advent of next generation sequencing technology in 2008, an increasing number of scientists use this technology to measure and understand changes in gene expression in often complex systems. As sequencing costs have decreased, using RNA-Seq to simultaneously measure the expression of tens of thousands of genes for multiple samples has never been easier. The cost of these experiments has now moved from generating the data to storing and analysing it.

There are many steps involved in analysing an RNA-Seq experiment. The analysis begins with sequencing reads (FASTQ files). These are usually aligned to a reference genome, if available. Then the number of reads mapped to each gene can be counted. This results in a table of counts, which is what we perform statistical analyses on to determine differentially expressed genes and pathways. The purpose of this tutorial is to demonstrate how to do read alignment and counting, prior to performing differential expression. Differential expression analysis with limma-voom is covered in an accompanying tutorial RNA-seq counts to genes. The tutorial here shows how to start from FASTQ data and perform the mapping and counting steps, along with associated Quality Control.

Mouse mammary gland dataset

The data for this tutorial comes from a Nature Cell Biology paper by Fu *et al.* 2015. Both the raw data (sequence reads) and processed data (counts) can be downloaded from Gene Expression Omnibus database (GEO) under accession number GSE60450.

This study examined the expression profiles of basal and luminal cells in the mammary gland of virgin, pregnant and lactating mice. Six groups are present, with one for each combination of cell type and mouse status. Note that two biological replicates are used here, two independent sorts of cells from the mammary glands of virgin, pregnant or lactating mice, however three replicates is usually recommended as a minimum requirement for RNA-seq.

This is a Galaxy tutorial based on material from the COMBINE R RNAseq workshop, first taught at a workshop in 2016.

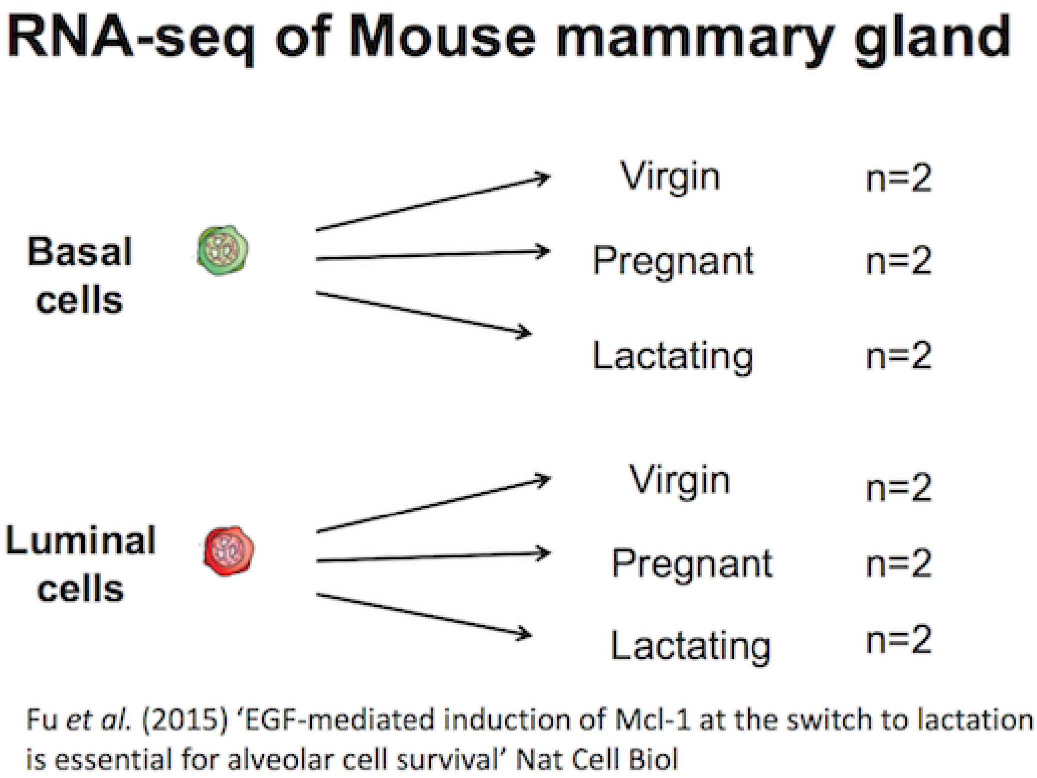

RNA-seq of Mouse mammary gland
Basal cells
Virgin n=2
Pregnant n=2
Lactating n=2
Luminal cells
Virgin n=2
Pregnant n=2
Lactating n=2
Fu *et al.* (2015) 'EGF-mediated induction of Mcl-1 at the switch to lactation is essential for alveolar cell survival' Nat Cell Biol

Figure 1: Tutorial Dataset

Agenda

In this tutorial, we will cover:

💬 Comment: Results may vary

Your results may be slightly different from the ones presented in this tutorial due to differing versions of tools, reference data, external databases, or because of stochastic processes in the algorithms.

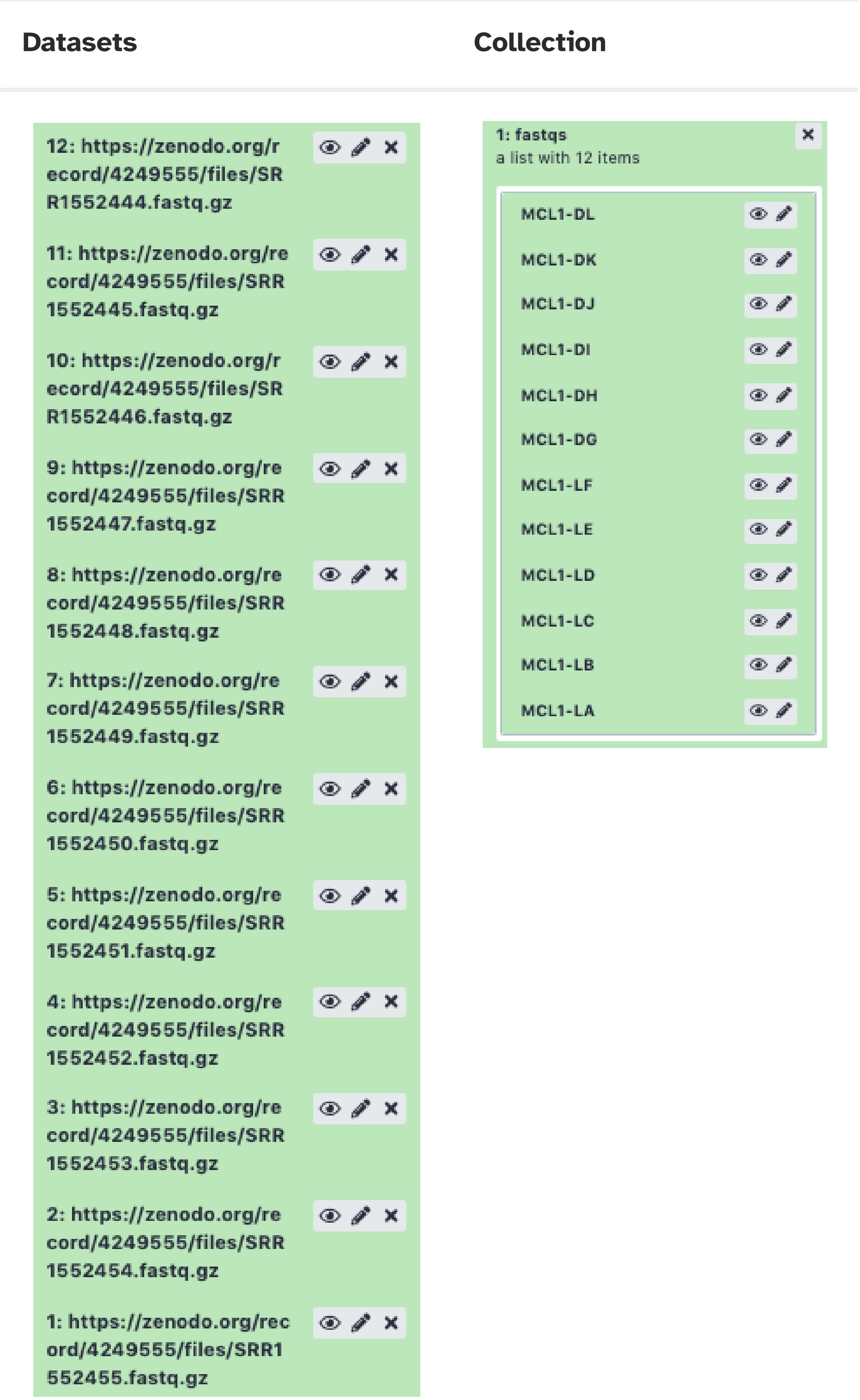
Preparing the reads

Import data from URLs

Read sequences are usually stored in compressed (gzipped) FASTQ files. Before the differential expression analysis can proceed, these reads must be aligned to the reference genome and counted into annotated genes. Mapping reads to the genome is a very important task, and many different aligners are available, such as HISAT2 (Kim *et al.* 2015), STAR (Dobin *et al.* 2013) and Subread (Liao *et al.* 2013). Most mapping tasks require larger computers than an average laptop, so usually read mapping is done on a server in a linux-like environment, requiring some programming knowledge. However, Galaxy enables you to do this mapping without needing to know programming and if you don’t have access to a server you can try to use one of the publically available Galaxies e.g. usegalaxy.org, usegalaxy.eu, usegalaxy.org.au.

The raw reads used in this tutorial were obtained from SRA from the link in GEO for the the mouse mammary gland dataset (e.g `ftp://ftp-trace.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByStudy/sra/SRP%2FSRP045%2FSRP045534`). For the purpose of this tutorial we are going to be working with a small part of the FASTQ files. We are only going to be mapping 1000 reads from each sample to enable running through all the steps quickly. If working with your own data you would use the full data and some results for the full mouse dataset will be shown for comparison. The small FASTQ files are available in Zenodo and the links to the FASTQ files are provided below.

If you are sequencing your own data, the sequencing facility will almost always provide compressed FASTQ files which you can upload into Galaxy. For sequence data available through URLs, The Galaxy Rule-based Uploader can be used to import the files. It is much quicker than downloading FASTQs to your computer and uploading into Galaxy and also enables importing as a **Collection**. When you have more than a few files, using Galaxy Collections helps keep the datasets organised and tidy in the history. Collections also make it easier to maintain the sample names through tools and workflows. If you are not familiar with collections, you can take a look at the Galaxy Collections tutorial for more details. The screenshots below show a comparison of what the FASTQ datasets for this tutorial would look like in the history if we imported them as datasets versus as a collection with the Rule-based Uploader.



Datasets

Collection

12: https://zenodo.org/record/4249555/files/SRR1552444.fastq.gz

11: https://zenodo.org/record/4249555/files/SRR1552445.fastq.gz

10: https://zenodo.org/record/4249555/files/SRR1552446.fastq.gz

9: https://zenodo.org/record/4249555/files/SRR1552447.fastq.gz

8: https://zenodo.org/record/4249555/files/SRR1552448.fastq.gz

7: https://zenodo.org/record/4249555/files/SRR1552449.fastq.gz

6: https://zenodo.org/record/4249555/files/SRR1552450.fastq.gz

5: https://zenodo.org/record/4249555/files/SRR1552451.fastq.gz

4: https://zenodo.org/record/4249555/files/SRR1552452.fastq.gz

3: https://zenodo.org/record/4249555/files/SRR1552453.fastq.gz

2: https://zenodo.org/record/4249555/files/SRR1552454.fastq.gz

1: https://zenodo.org/record/4249555/files/SRR1552455.fastq.gz

1: fastqs
a list with 12 items

MCL1-DL
MCL1-DK
MCL1-DJ
MCL1-DI
MCL1-DH
MCL1-DG
MCL1-LF
MCL1-LE
MCL1-LD
MCL1-LC
MCL1-LB
MCL1-LA

Details: Collections and sample names

The information we need to import the samples for this tutorial (sample ID, Group, and link to the FASTQ file (URL) are in the grey box below.

```
SampleID        Group   URL
MCL1-DL basallactate    https://zenodo.org/record/4249555/files/SRR1552455.fastq.gz
MCL1-DK basallactate    https://zenodo.org/record/4249555/files/SRR1552454.fastq.gz
MCL1-DJ basalpregnant   https://zenodo.org/record/4249555/files/SRR1552453.fastq.gz
MCL1-DI basalpregnant   https://zenodo.org/record/4249555/files/SRR1552452.fastq.gz
MCL1-DH basalvirgin     https://zenodo.org/record/4249555/files/SRR1552451.fastq.gz
MCL1-DG basalvirgin     https://zenodo.org/record/4249555/files/SRR1552450.fastq.gz
MCL1-LF luminallactate  https://zenodo.org/record/4249555/files/SRR1552449.fastq.gz
MCL1-LE luminallactate  https://zenodo.org/record/4249555/files/SRR1552448.fastq.gz
MCL1-LD luminalpregnant https://zenodo.org/record/4249555/files/SRR1552447.fastq.gz
MCL1-LC luminalpregnant https://zenodo.org/record/4249555/files/SRR1552446.fastq.gz
MCL1-LB luminalvirgin   https://zenodo.org/record/4249555/files/SRR1552445.fastq.gz
MCL1-LA luminalvirgin   https://zenodo.org/record/4249555/files/SRR1552444.fastq.gz
```

In order to get these files into Galaxy, we will want to do a few things:

- Strip the *header* out of the sample information (it doesn't contain a URL Galaxy can download).
- Define the file **Identifier** column (`SampleID`).
- Define the **URL** column (`URL`) (this is the location Galaxy can download the data from).

Hands-on: Data upload

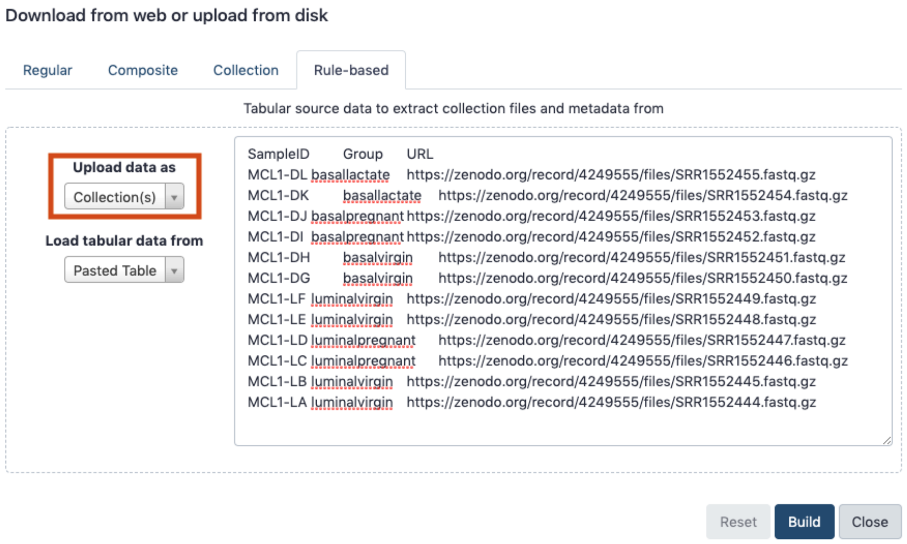
1. Create a new history for this tutorial e.g. `RNA-seq reads to counts`

   > Tip: Creating a new history ➕

   > Tip: Renaming a history ➕

2. Import the files from Zenodo using Galaxy's Rule-based Uploader.

   - Open the Galaxy Upload Manager
   - Click the tab **Rule-based**
     - *"Upload data as"*: `Collection(s)`
     - *"Load tabular data from"*: `Pasted Table`
   - Paste the table from the grey box above. *(You should now see below)*
   - Click **Build**



Tabular source data to extract collection files and metadata from

Upload data as: Collection(s)

Load tabular data from: Pasted Table

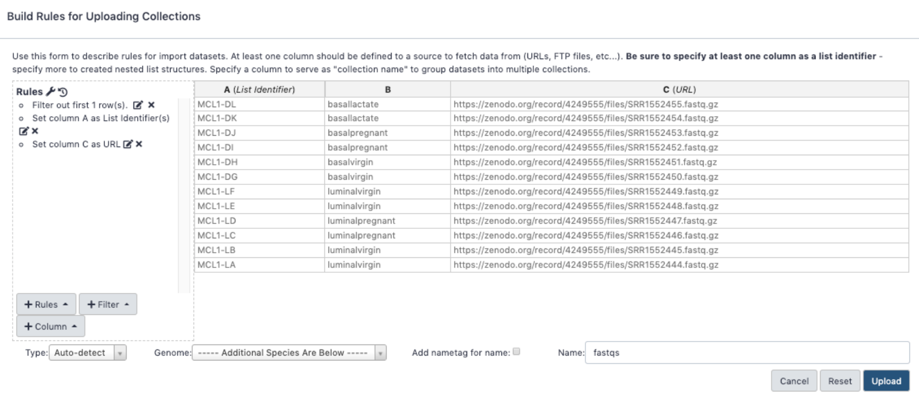
SampleID	Group	URL
MCL1-DL	basallactate	https://zenodo.org/record/4249555/files/SRR1552455.fastq.gz
MCL1-DK	basallactate	https://zenodo.org/record/4249555/files/SRR1552454.fastq.gz
MCL1-DJ	basalpregnant	https://zenodo.org/record/4249555/files/SRR1552453.fastq.gz
MCL1-DI	basalpregnant	https://zenodo.org/record/4249555/files/SRR1552452.fastq.gz
MCL1-DH	basalvirgin	https://zenodo.org/record/4249555/files/SRR1552451.fastq.gz
MCL1-DG	basalvirgin	https://zenodo.org/record/4249555/files/SRR1552450.fastq.gz
MCL1-LF	luminalvirgin	https://zenodo.org/record/4249555/files/SRR1552449.fastq.gz
MCL1-LE	luminalvirgin	https://zenodo.org/record/4249555/files/SRR1552448.fastq.gz
MCL1-LD	luminalpregnant	https://zenodo.org/record/4249555/files/SRR1552447.fastq.gz
MCL1-LC	luminalpregnant	https://zenodo.org/record/4249555/files/SRR1552446.fastq.gz
MCL1-LB	luminalvirgin	https://zenodo.org/record/4249555/files/SRR1552445.fastq.gz
MCL1-LA	luminalvirgin	https://zenodo.org/record/4249555/files/SRR1552444.fastq.gz

Figure 2: Rule-based Uploader

   - In the `rules editor` that pops up:
     - **Remove the header**. From the **Filter** menu select `First or Last N Rows`
       - *"Filter which rows?"*: `first`
       - *"Filter how many rows?"*: `1`
       - Click `Apply`
     - **Define the Identifier and URL columns**. From the **Rules** menu select `Add / Modify Column Definitions`
       - Click `Add Definition` button and select List Identifier(s)
         - *"List Identifier(s)"*: `A`
       - Click `Add Definition` button again and select URL instead
         - *"URL"*: `C`

- Click `Apply`, and you should see your new column definitions listed

- **Name the collection**. For *“Name”* enter: `fastqs` *(You should now see below)*
- Click `Upload`



Build Rules for Uploading Collections

Use this form to describe rules for import datasets. At least one column should be defined to a source to fetch data from (URLs, FTP files, etc...). **Be sure to specify at least one column as a list identifier** - specify more to created nested list structures. Specify a column to serve as "collection name" to group datasets into multiple collections.

Rules
- Filter out first 1 row(s).
- Set column A as List Identifier(s)
- Set column C as URL

A (*List Identifier*)	B	C (*URL*)
MCL1-DL	basallactate	https://zenodo.org/record/4249555/files/SRR1552455.fastq.gz
MCL1-DK	basallactate	https://zenodo.org/record/4249555/files/SRR1552454.fastq.gz
MCL1-DJ	basalpregnant	https://zenodo.org/record/4249555/files/SRR1552453.fastq.gz
MCL1-DI	basalpregnant	https://zenodo.org/record/4249555/files/SRR1552452.fastq.gz
MCL1-DH	basalvirgin	https://zenodo.org/record/4249555/files/SRR1552451.fastq.gz
MCL1-DG	basalvirgin	https://zenodo.org/record/4249555/files/SRR1552450.fastq.gz
MCL1-LF	luminalvirgin	https://zenodo.org/record/4249555/files/SRR1552449.fastq.gz
MCL1-LE	luminalvirgin	https://zenodo.org/record/4249555/files/SRR1552448.fastq.gz
MCL1-LD	luminalpregnant	https://zenodo.org/record/4249555/files/SRR1552447.fastq.gz
MCL1-LC	luminalpregnant	https://zenodo.org/record/4249555/files/SRR1552446.fastq.gz
MCL1-LB	luminalvirgin	https://zenodo.org/record/4249555/files/SRR1552445.fastq.gz
MCL1-LA	luminalvirgin	https://zenodo.org/record/4249555/files/SRR1552444.fastq.gz

Type: Auto-detect Genome: ----- Additional Species Are Below ----- Add nametag for name: □ Name: fastqs

Figure 3: Rules Editor

You should see a collection (list) called `fastqs` in your history containing all 12 FASTQ files.

Link to here | FAQs | Gitter Chat | Help Forum

If your data is not accessible by URL, for example, if your FASTQ files are located on your laptop and are not too large, you can upload into a collection as below. If they are large you could use FTP. You can take a look at the Getting data into Galaxy slides for more information.

Tip: Upload local files into a collection

If your FASTQ files are located in Shared Data, you can import them into your history as a collection as below.

Tip: Import files from Shared Data into a collection

Take a look at one of the FASTQ files to see what it contains.

Hands-on: Take a look at FASTQ format

1. Click on the collection name (`fastqs`)
2. Click on the (eye) icon of one of the FASTQ files to have a look at what it contains

Link to here | FAQs | Gitter Chat | Help Forum

Details: FASTQ format

Raw reads QC

During sequencing, errors are introduced, such as incorrect nucleotides being called. These are due to the technical limitations of each sequencing platform. Sequencing errors might bias the analysis and can lead to a misinterpretation of the data. Every base sequence gets a quality score from the sequencer and this information is present in the FASTQ file. A quality score of 30 corresponds to a 1 in 1000 chance of an incorrect base call (a quality score of 10 is a 1 in 10 chance of an incorrect base call). To look at the overall distribution of quality scores across the reads, we can use FastQC.

Sequence quality control is therefore an essential first step in your analysis. We will use similar tools as described in the "Quality control" tutorial: FastQC and Cutadapt (Marcel 2011).

✎ Hands-on: Check raw reads with **FastQC**

1. **FastQC** 🔧⚙
   - 🗀 *"Short read data from your current history"*: `fastqs` (Input dataset collection)
2. Inspect the `Webpage` output of **FastQC** 🔧 for the `MCL1-DL` sample by clicking on the 👁 (eye) icon

💡 Tip: Selecting a dataset collection as input ⊞

Link to here | ⓘ FAQs | Gitter Chat | Help Forum

ⓘ Question

1. What is the read length?
2. What base quality score encoding is used?

👁 Solution ⊞

The FastQC report contains a lot of information and we can look at the report for each sample. However, that is quite a few reports, 12 for this dataset. If you had more samples it could be a lot more. Luckily, there is a very useful tool called MultiQC (Ewels *et al.* 2016) that can summarise QC information for multiple samples into a single report. We'll generate a few MultiQC outputs in this tutorial so we'll add name tags so we can differentiate them.

✎ Hands-on: Aggregate FastQC reports with **MultiQC**

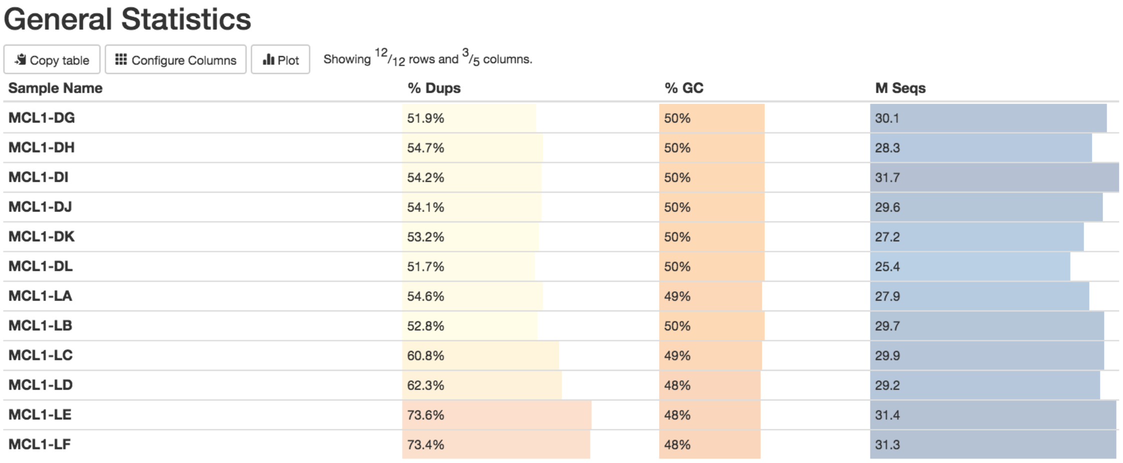
1. **MultiQC** with the following parameters to aggregate the FastQC reports
   - In *"Results"*
     - *"Which tool was used generate logs?"*: `FastQC`
     - In *"FastQC output"*
       - *"Type of FastQC output?"*: `Raw data`
       - *"FastQC output"*: `RawData` files (output of **FastQC** on trimmed reads)
2. Add a tag `#fastqc-raw` to the `Webpage` output from MultiQC and inspect the webpage

Tip: Adding a tag

Link to here | FAQs | Gitter Chat | Help Forum

Note that these are the results for just 1000 reads. The FastQC results for the full dataset are shown below. The 1000 reads are the first reads from the FASTQ files, and the first reads usually originate from the flowcell edges, so we can expect that they may have lower quality and the patterns may be a bit different from the distribution in the full dataset.

You should see that most of the plots in the small FASTQs look similar to the full dataset. However, in the small FASTQs, there is less duplication, some Ns in the reads and some overrepresented sequences.



General Statistics

Showing 12/12 rows and 3/5 columns.

Sample Name	% Dups	% GC	M Seqs
MCL1-DG	51.9%	50%	30.1
MCL1-DH	54.7%	50%	28.3
MCL1-DI	54.2%	50%	31.7
MCL1-DJ	54.1%	50%	29.6
MCL1-DK	53.2%	50%	27.2
MCL1-DL	51.7%	50%	25.4
MCL1-LA	54.6%	49%	27.9
MCL1-LB	52.8%	50%	29.7
MCL1-LC	60.8%	49%	29.9
MCL1-LD	62.3%	48%	29.2
MCL1-LE	73.6%	48%	31.4
MCL1-LF	73.4%	48%	31.3

Figure 5: General Statistics

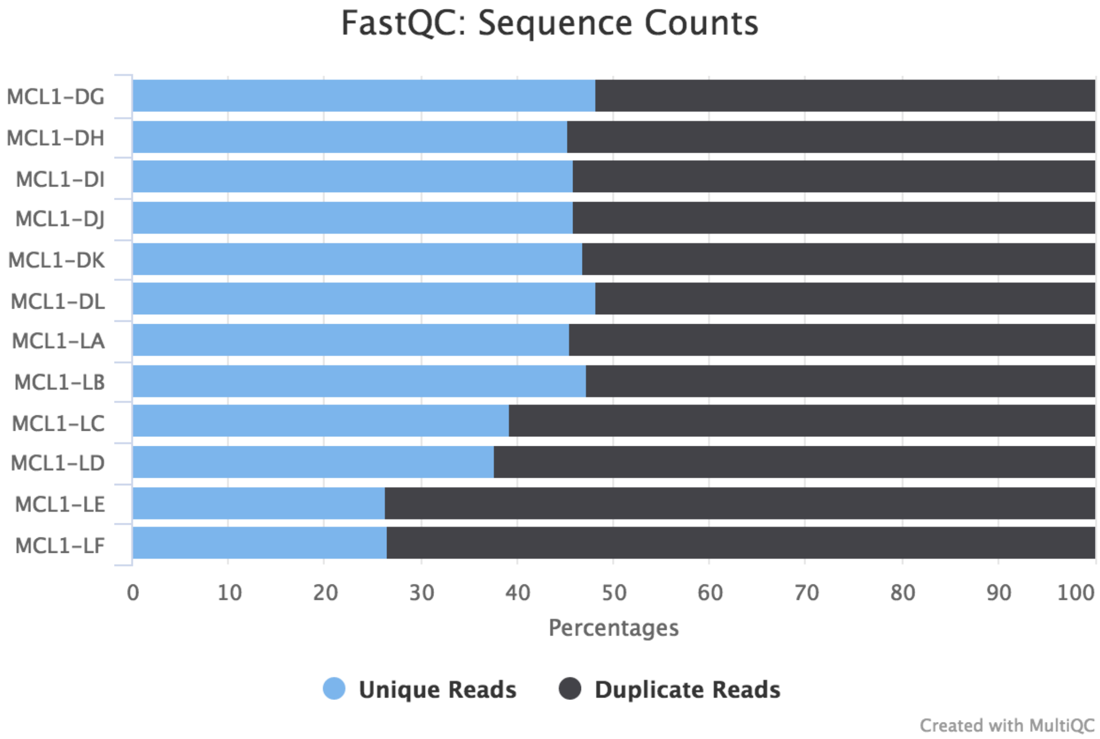


Figure 6: Sequence Counts

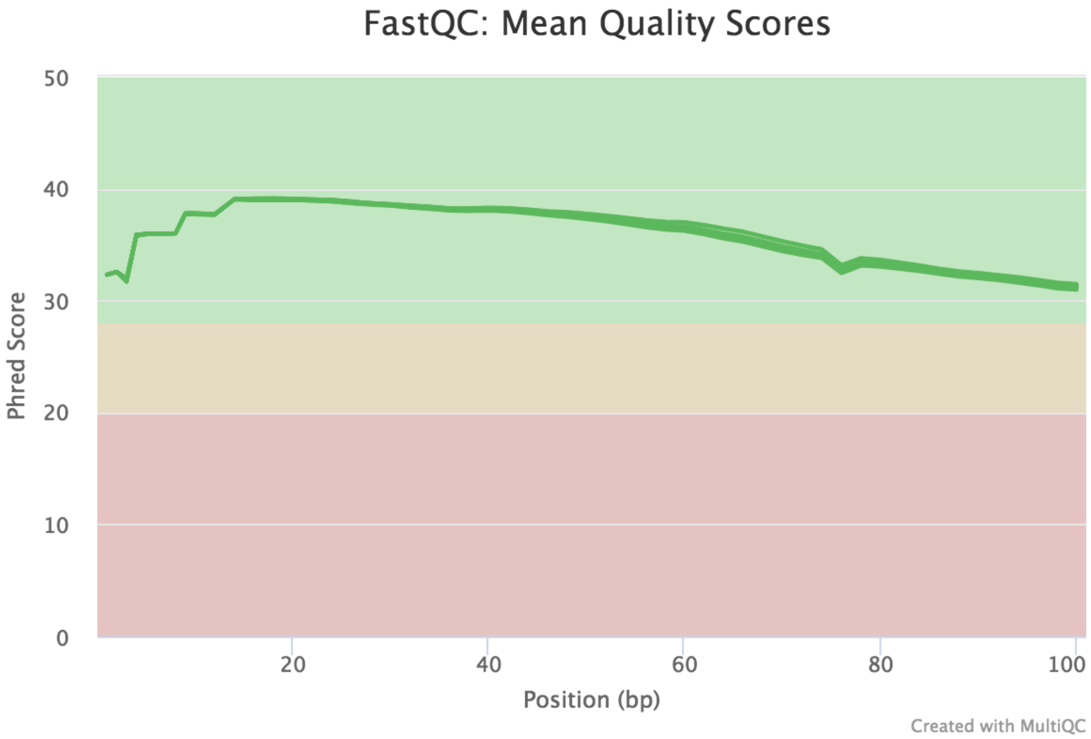


Figure 7: Sequence Quality

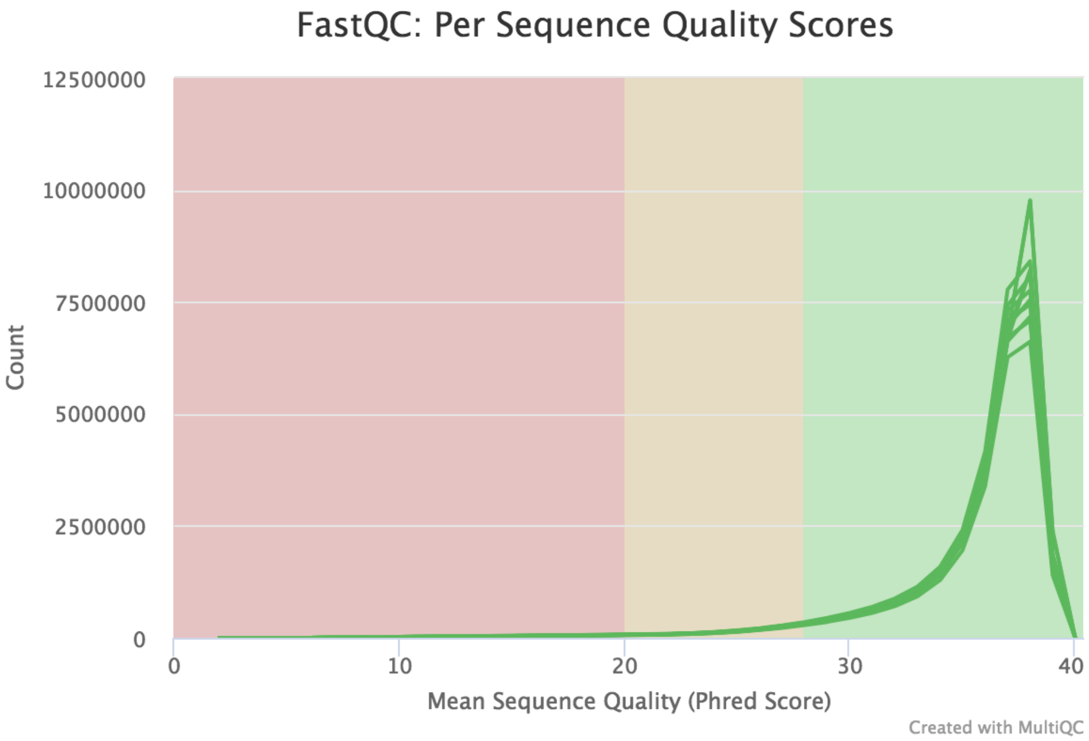


Figure 8: Per Sequence Quality Scores

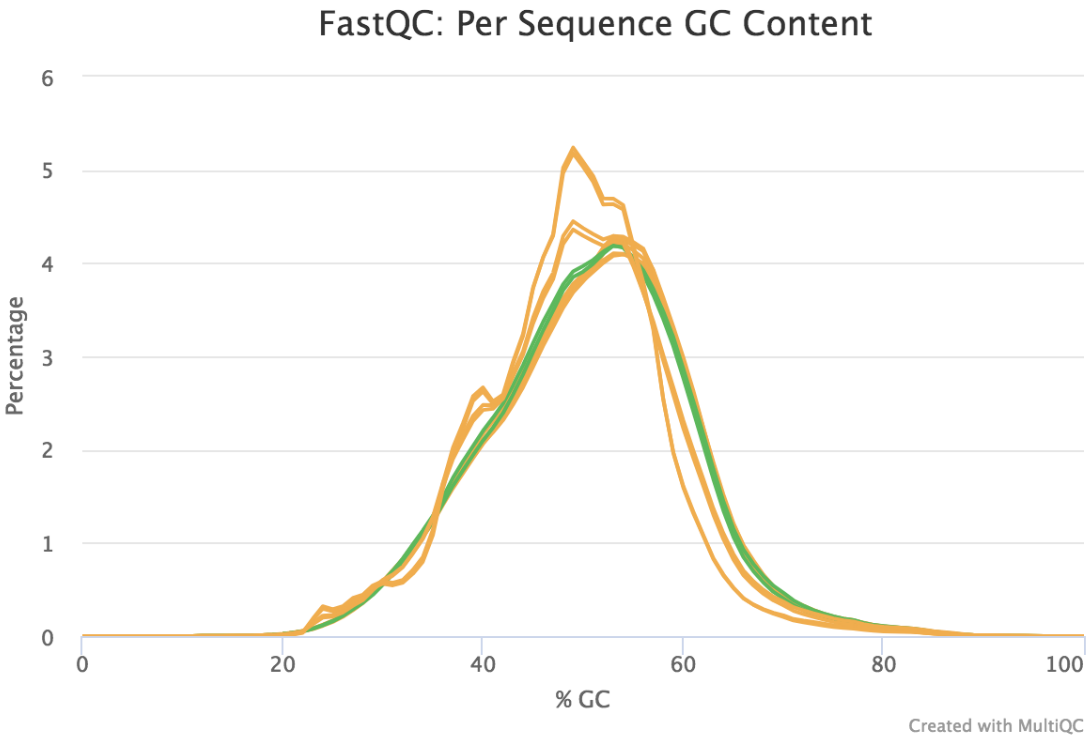


Figure 9: Per Sequence GC Content

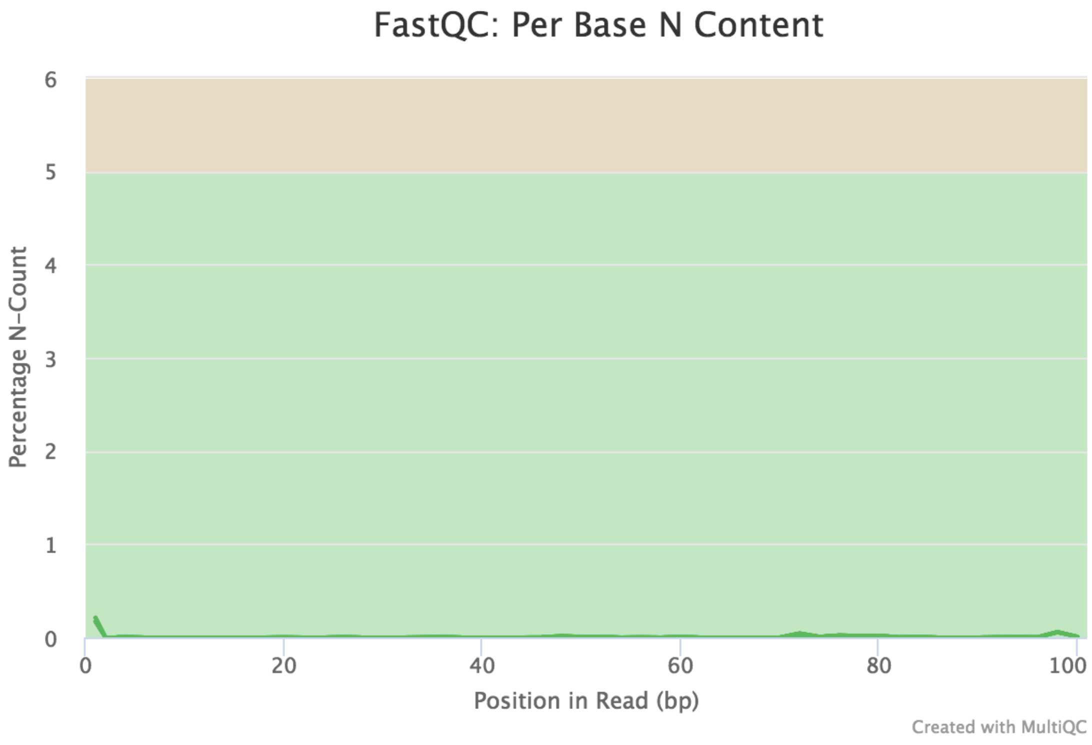


Figure 10: Per base N content

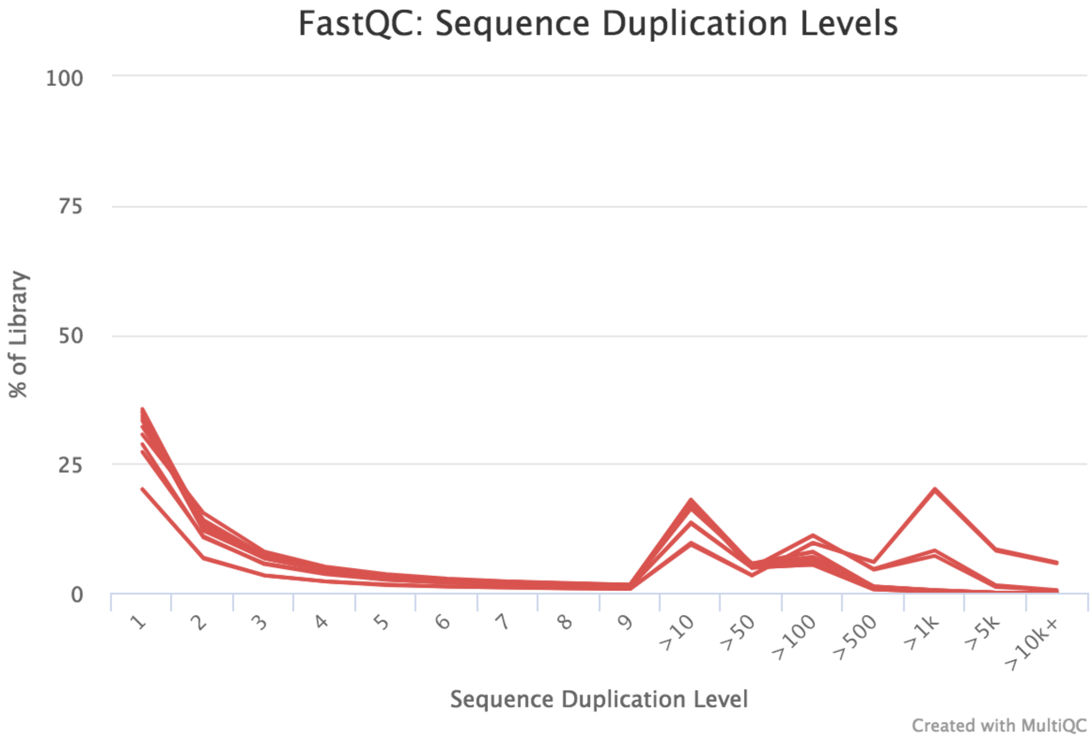


Figure 11: Sequence Duplication Levels

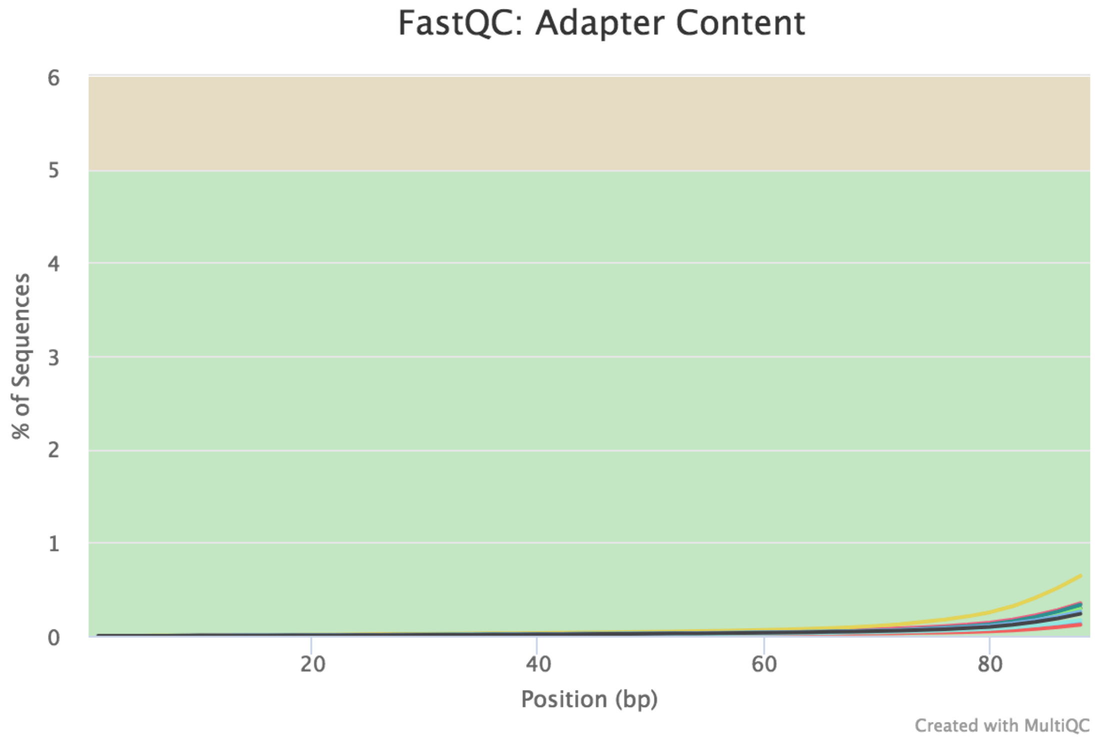


Figure 12: Adapter Content

See the Quality Control tutorial for more information on FastQC plots.

Question

What do you think of the overall quality of the sequences?

Solution

We will use Cutadapt to trim the reads to remove the Illumina adapter and any low quality bases at the ends (quality score < 20). We will discard any sequences that are too short (< 20bp) after trimming. We will also output the Cutadapt report for summarising with MultiQC.

The Cutadapt tool Help section provides the sequence we can use to trim this standard Illumina adapter `AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC`, as given on the Cutadapt website. For trimming paired-end data see the Cutadapt Help section. Other Illumina adapter sequences (e.g. Nextera) can be found at the Illumina website. Note that Cutadapt requires at least three bases to match between adapter and read to reduce the number of falsely trimmed bases, which can be changed in the Cutadapt options if desired.

Trim reads

Hands-on: Trim reads with **Cutadapt**

1. **Cutadapt**
   - *“Single-end or Paired-end reads?”*: `Single-end`
     - *“FASTQ/A file”*: `fastqs` (Input dataset collection)
     - In *“Read 1 Options”*:
       - In *“3’ (End) Adapters”*:
         - Click on *“Insert 3’ (End) Adapters”*:
         - In *“1: 3’ (End) Adapters”*:
           - *“Source”*: `Enter custom sequence`
             - *“Enter custom 3’ adapter name (Optional)”*: `Illumina`
             - *“Enter custom 3’ adapter sequence”*: `AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC`
   - In *“Filter Options”*:
     - *“Minimum length (R1)”*: `20`
   - In *“Read Modification Options”*:
     - *“Quality cutoff”*: `20`
   - ☑ *“Outputs selector”*: `Report`

Link to here | FAQs | Gitter Chat | Help Forum

We can take a look at the reads again now that they’ve been trimmed.

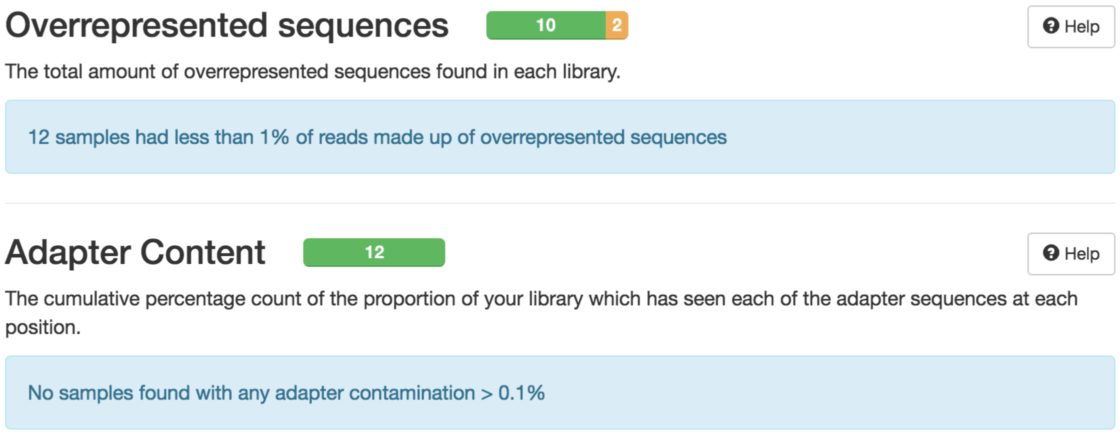
Trimmed reads QC

Hands-on: QC of trimmed reads with **FastQC**

1. **FastQC**
   - *“Short read data from your current history”*: `RawData` (output of **Cutadapt**)
2. **MultiQC** with the following parameters to aggregate the FastQC reports
   - In *“Results”*
     - *“Which tool was used generate logs?”*: `FastQC`
     - In *“FastQC output”*
       - *“Type of FastQC output?”*: `Raw data`
       - *“FastQC output”*: `RawData` files (output of **FastQC**)
3. Add a tag `#fastqc-trimmed` to the `Webpage` output from MultiQC and inspect the webpage

Link to here | FAQs | Gitter Chat | Help Forum

The MultiQC plot below shows the result from the full dataset for comparison.



Overrepresented sequences

The total amount of overrepresented sequences found in each library.

12 samples had less than 1% of reads made up of overrepresented sequences

Adapter Content

The cumulative percentage count of the proportion of your library which has seen each of the adapter sequences at each position.

No samples found with any adapter contamination > 0.1%

Figure 13: Adapter Content post-trimming

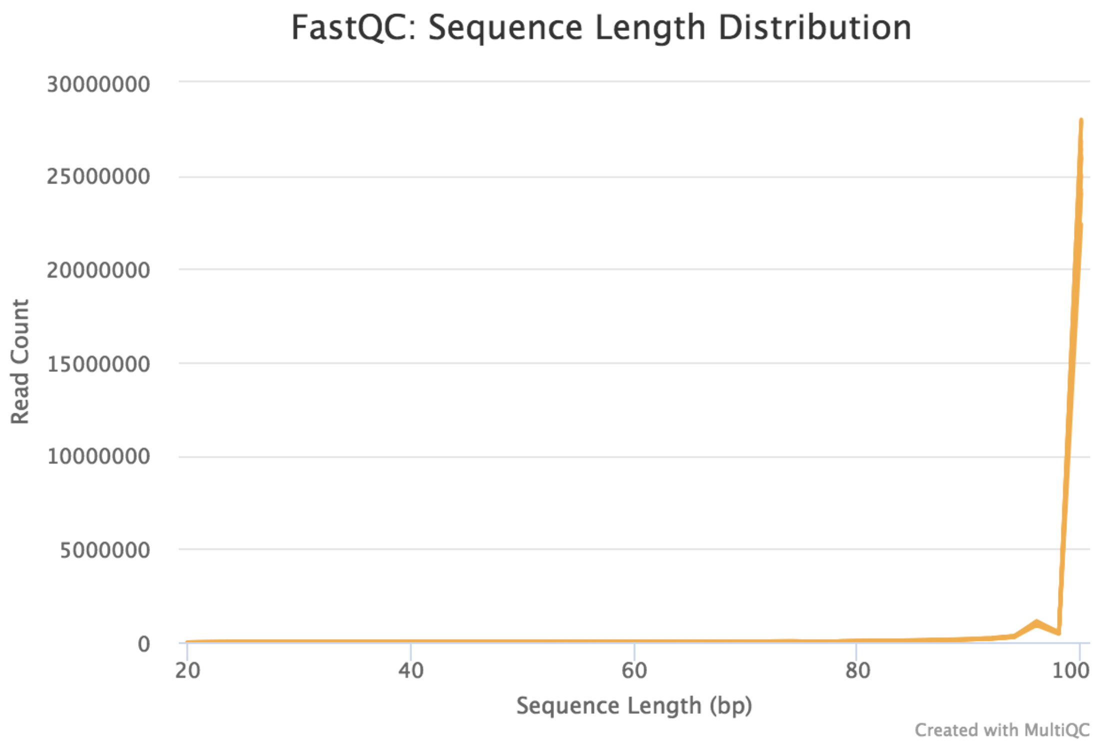


Figure 14: Sequence Length post-trimming

After trimming we can see that:

- No adapter is detected now.
- The reads are no longer all the same length, we now have sequences of different lengths detected.

Mapping

Now that we have prepared our reads, we can align the reads for our 12 samples. There is an existing reference genome for mouse and we will map the reads to that. The current most widely used version of the mouse reference genome is `mm10/GRCm38` (although note that there is a new version `mm39` released June 2020). Here we will use **HISAT2** to align the reads. HISAT2 is the descendent of TopHat, one of the first widely-used aligners, but alternative mappers could be used, such as STAR. See the RNA-seq ref-based tutorial for more information on RNA-seq mappers. There are often numerous mapping parameters that we can specify, but usually the default mapping parameters are fine. However, library type (paired-end vs single-end) and library strandness (stranded vs unstranded) require some different settings when mapping and counting, so they are two important pieces of information to know about samples. The mouse data comprises unstranded, single-end reads so we will specify that where necessary. HISAT2 can output a mapping summary file that tells what proportion of reads mapped to the reference genome. Summary files for multiple samples can be summarised with MultiQC. As we're only using a subset of 1000 reads per sample, aligning should just take a minute or so. To run the full samples from this dataset would take longer.

Map reads to reference genome

Hands-on: Map reads to reference with **HISAT2**

1. **HISAT2** with the following parameters:
   - *"Source for the reference genome"*: `Use a built-in genome`
     - *"Select a reference genome"*: `mm10`
   - *"Is this a single or paired library?"*: `Single-end`
     - *"FASTA/Q file"*: `Read 1 Output` (output of **Cutadapt**)
   - In *"Summary Options"*:
     - ☑ *"Output alignment summary in a more machine-friendly style."*: `Yes`
     - ☑ *"Print alignment summary to a file."*: `Yes`
2. **MultiQC** with the following parameters to aggregate the HISAT2 summary files
   - In *"Results"*
     - *"Which tool was used generate logs?"*: `HISAT2`
     - *"Output of HISAT2"*: `Mapping summary` (output of **HISAT2**)
3. Add a tag `#hisat` to the `Webpage` output from MultiQC and inspect the webpage

Link to here | FAQs | Gitter Chat | Help Forum

Comment: Settings for Paired-end or Stranded reads

- If you have **paired-end** reads
  - Select *"Is this a single or paired library"* `Paired-end` or `Paired-end Dataset Collection` or `Paired-end data from single interleaved dataset`
- If you have **stranded** reads
  - Select *"Specify strand information"*: `Forward (FR)` or `Reverse (RF)`

The MultiQC plot below shows the result from the full dataset for comparison.

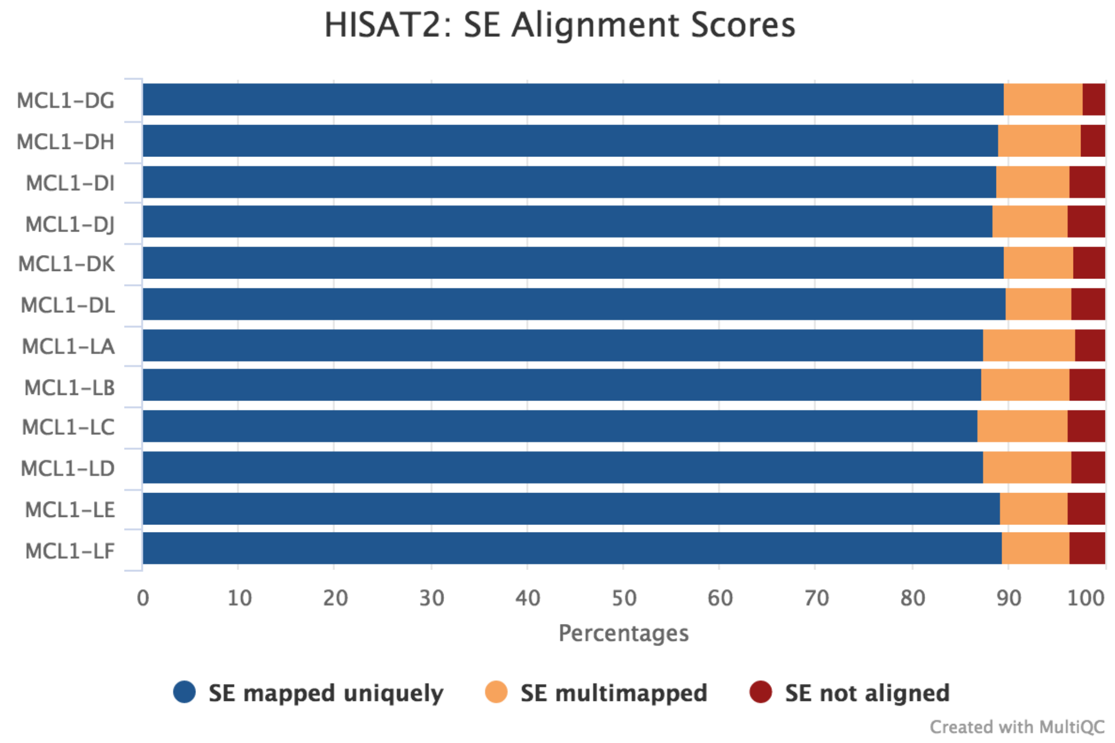


Figure 15: HISAT2 mapping

An important metric to check is the percentage of reads mapped to the reference genome. A low percentage can indicate issues with the data or analysis. Over 90% of reads have mapped in all samples, which is a good mapping rate, and the vast majority of reads have mapped uniquely, they haven't mapped to multiple locations in the reference genome.

It is also good practice to visualise the read alignments in the BAM file, for example using IGV, see the RNA-seq ref-based tutorial.

Link to here | FAQs | Gitter Chat | Help Forum

HISAT2 generates a BAM file with mapped reads.

A BAM (Binary Alignment Map) file is a compressed binary file storing the read sequences, whether they have been aligned to a reference sequence (e.g. a chromosome), and if so, the position on the reference sequence at which they have been aligned.

Hands-on: Inspect a BAM/SAM file

1. Inspect the output of **HISAT2**

Link to here | FAQs | Gitter Chat | Help Forum

A BAM file (or a SAM file, the non-compressed version) consists of:

- A header section (the lines starting with `@`) containing metadata particularly the chromosome names and lengths (lines starting with the `@SQ` symbol)
- An alignment section consisting of a table with 11 mandatory fields, as well as a variable number of optional fields:

Col	Field	Type	Brief Description
1	QNAME	String	Query template NAME
2	FLAG	Integer	Bitwise FLAG
3	RNAME	String	References sequence NAME
4	POS	Integer	1- based leftmost mapping POSition
5	MAPQ	Integer	MAPping Quality
6	CIGAR	String	CIGAR String
7	RNEXT	String	Ref. name of the mate/next read
8	PNEXT	Integer	Position of the mate/next read
9	TLEN	Integer	Observed Template LENgth
10	SEQ	String	Segment SEQuence
11	QUAL	String	ASCII of Phred-scaled base QUALity+33

Question

1. Which information do you find in a SAM/BAM file?
2. What is the additional information compared to a FASTQ file?

Solution

Tip: Downloading a collection

Counting

The alignment produces a set of BAM files, where each file contains the read alignments for each sample. In the BAM file, there is a chromosomal location for every read that mapped. Now that we have figured out where each read comes from in the genome, we need to summarise the information across genes or exons. The mapped reads can be counted across mouse genes by using a tool called featureCounts (Liao *et al.* 2013). featureCounts requires gene annotation specifying the genomic start and end position of each exon of each gene. For convenience, featureCounts contains built-in annotation for mouse (`mm10`, `mm9`) and human (`hg38`, `hg19`) genome assemblies, where exon intervals are defined from the NCBI RefSeq annotation of the reference genome. Reads that map to exons of genes are added together to obtain the count for each gene, with some care taken with reads that span exon-exon boundaries. The output is a count for each Entrez Gene ID, which are numbers such as `100008567`. For other species, users will need to read in a data frame in GTF format to define the genes and exons. Users can also specify a custom annotation file in SAF format. See the tool help in Galaxy, which has an example of what an SAF file should like like, or the Rsubread users guide for more information.

💬 Comment

In this example we have kept many of the default settings, which are typically optimised to work well under a variety of situations. For example, the default setting for featureCounts is that it only keeps reads that uniquely map to the reference genome. For testing differential expression of genes, this is preferred, as the reads are unambigously assigned to one place in the genome, allowing for easier interpretation of the results. Understanding all the different parameters you can change involves doing a lot of reading about the tool that you are using, and can take a lot of time to understand! We won't be going into the details of the parameters you can change here, but you can get more information from looking at the tool help.

Count reads mapped to genes

✏ Hands-on: Count reads mapped to genes with **featureCounts**

1. **featureCounts** 🔧⚙ with the following parameters:
   - 🗀 *"Alignment file"*: `aligned reads (BAM)` (output of **HISAT2** 🔧)
   - ▼ *"Gene annotation file"*: `featureCounts built-in`
     - ▼ *"Select built-in genome"*: `mm10`
2. **MultiQC** 🔧⚙ with the following parameters:
   - ▼ *"Which tool was used generate logs?"*: `featureCounts`
     - 🗀 *"Output of FeatureCounts"*: `featureCounts summary` (output of **featureCounts** 🔧)
3. Add a tag `#featurecounts` to the `Webpage` output from MultiQC and inspect the webpage

Link to here | ⓘ FAQs | Gitter Chat | Help Forum

💬 Comment: Settings for Paired-end or Stranded reads

- If you have **paired-end** reads
  - Click *"Options for paired-end reads"*
    - *"Count fragments instead of reads"*: `Enabled; fragments (or templates) will be counted instead of reads`
- If you have **stranded** reads
  - Select *"Specify strand information"*: `Stranded (Forward)` or `Stranded (Reverse)`

The MultiQC plot below shows the result from the full dataset for comparison.

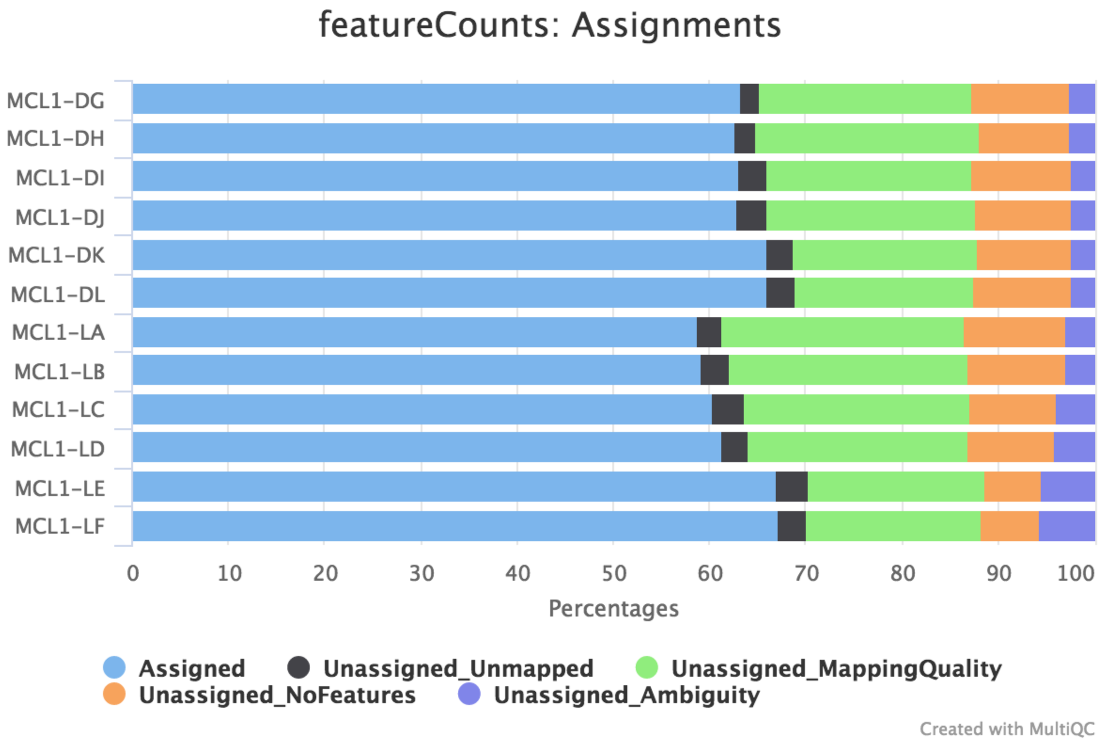


Figure 16: featureCounts assignments

Question

What % reads are assigned to exons?

Solution

The counts for the samples are output as tabular files. Take a look at one. The numbers in the first column of the counts file represent the Entrez gene identifiers for each gene, while the second column contains the counts for each gene for the sample.

Create count matrix

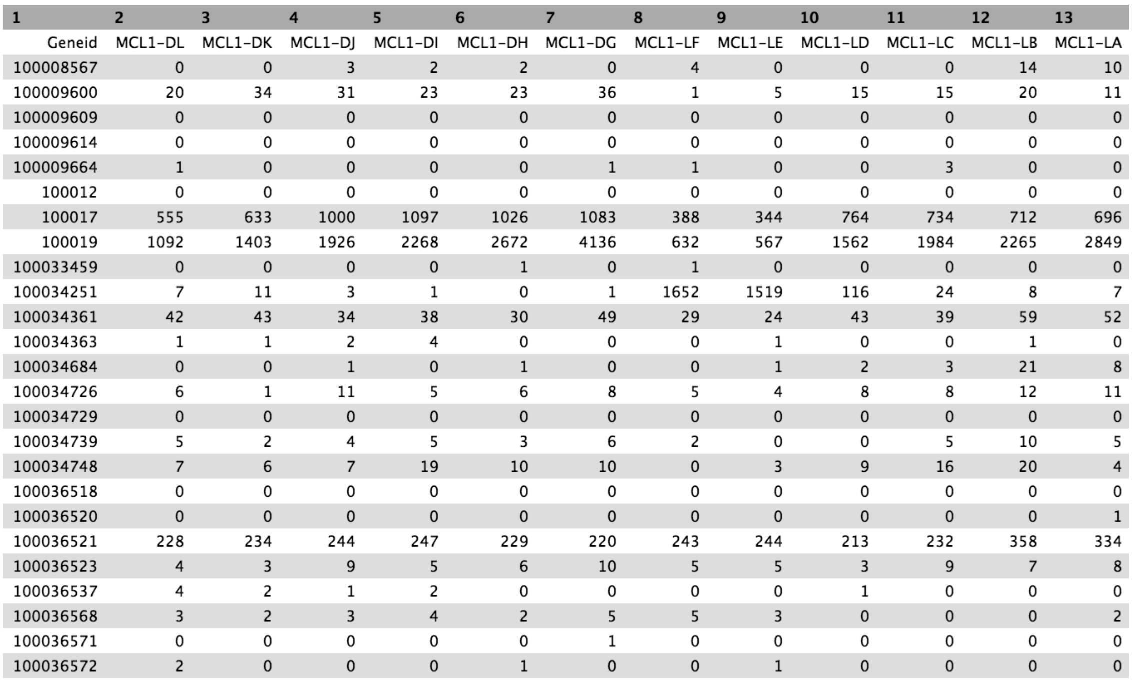
The counts files are currently in the format of one file per sample. However, it is often convenient to have a count matrix. A count matrix is a single table containing the counts for all samples, with the genes in rows and the samples in columns. The counts files are all within a collection so we can use the Galaxy **Column Join on multiple datasets** tool to easily create a count matrix from the single counts files.

Hands-on: Create count matrix with **Column Join on multiple datasets**

1. **Column Join on multiple datasets** with the following parameters:
   - *"Tabular files"*: `Counts` (output of **featureCounts**)
   - *"Identifier column"*: `1`
   - *"Number of header lines in each input file"*: `1`
   - ☑ *"Add column name to header"*: `No`

Link to here | FAQs | Gitter Chat | Help Forum

Take a look at the output. Note that as the tutorial uses a small subset of the data (~ `1000` reads per sample), to save on processing time, most rows in that matrix will contain all zeros (there will be ~`600` non-zero rows). The output for the full dataset is shown below.



1	2	3	4	5	6	7	8	9	10	11	12	13
Geneid	MCL1-DL	MCL1-DK	MCL1-DJ	MCL1-DI	MCL1-DH	MCL1-DG	MCL1-LF	MCL1-LE	MCL1-LD	MCL1-LC	MCL1-LB	MCL1-LA
100008567	0	0	3	2	2	0	4	0	0	0	14	10
100009600	20	34	31	23	23	36	1	5	15	15	20	11
100009609	0	0	0	0	0	0	0	0	0	0	0	0
100009614	0	0	0	0	0	0	0	0	0	0	0	0
100009664	1	0	0	0	0	1	1	0	0	3	0	0
100012	0	0	0	0	0	0	0	0	0	0	0	0
100017	555	633	1000	1097	1026	1083	388	344	764	734	712	696
100019	1092	1403	1926	2268	2672	4136	632	567	1562	1984	2265	2849
100033459	0	0	0	0	1	0	1	0	0	0	0	0
100034251	7	11	3	1	0	1	1652	1519	116	24	8	7
100034361	42	43	34	38	30	49	29	24	43	39	59	52
100034363	1	1	2	4	0	0	0	1	0	0	1	0
100034684	0	0	1	0	1	0	0	1	2	3	21	8
100034726	6	1	11	5	6	8	5	4	8	8	12	11
100034729	0	0	0	0	0	0	0	0	0	0	0	0
100034739	5	2	4	5	3	6	2	0	0	5	10	5
100034748	7	6	7	19	10	10	0	3	9	16	20	4
100036518	0	0	0	0	0	0	0	0	0	0	0	0
100036520	0	0	0	0	0	0	0	0	0	0	0	1
100036521	228	234	244	247	229	220	243	244	213	232	358	334
100036523	4	3	9	5	6	10	5	5	3	9	7	8
100036537	4	2	1	2	0	0	0	0	1	0	0	0
100036568	3	2	3	4	2	5	5	3	0	0	0	2
100036571	0	0	0	0	0	1	0	0	0	0	0	0
100036572	2	0	0	0	1	0	0	1	0	0	0	0

Figure 17: Count matrix

Now it is easier to see the counts for a gene across all samples. The accompanying tutorial, RNA-seq counts to genes, shows how gene information (symbols etc) can be added to a count matrix.

Generating a QC summary report

There are several additional QCs we can perform to better understand the data, to see if it’s good quality. These can also help determine if changes could be made in the lab to improve the quality of future datasets.

We’ll use a prepared workflow to run the first few of the QCs below. This will also demonstrate how you can make use of Galaxy workflows to easily run and reuse multiple analysis steps. The workflow will run the first three tools: **Infer Experiment**, **MarkDuplicates** and **IdxStats** and generate a **MultiQC** report. You can then edit the workflow if you’d like to add other steps.

Hands-on: Run QC report workflow

1. **Import the workflow** into Galaxy
   - Copy the URL (e.g. via right-click) of this workflow or download it to your computer.
   - Import the workflow into Galaxy

   Tip: Importing a workflow ⊞

2. Import this file as type BED file:

   ```
   https://sourceforge.net/projects/rseqc/files/BED/Mouse_Mus_musculus/mm10_RefSeq.bed.gz/download
   ```

   Tip: Importing via links ⊞

3. Run **Workflow QC Report** using the following parameters:
   - *“Send results to a new history”*: `No`
   - *“1: Reference genes”*: the imported RefSeq BED file
   - *“2: BAM files”*: `aligned reads (BAM)` (output of **HISAT2**)

   Tip: Running a workflow ⊞

4. Inspect the `Webpage` output from MultiQC

Link to here | FAQs | Gitter Chat | Help Forum

You do not need to run the hands-on steps below. They are just to show how you could run the tools individually and what parameters to set.

Strandness

As far as we know this data is unstranded, but as a sanity check you can check the strandness. You can use RSeQC Infer Experiment tool to "guess" the strandness, as explained in the RNA-seq ref-based tutorial. This is done through comparing the "strandness of reads" with the "strandness of transcripts". For this tool, and many of the other RSeQC (Wang *et al.* 2012) tools, a reference bed file of genes (`reference genes`) is required. RSeQC provides some reference BED files for model organisms. You can import the RSeQC mm10 RefSeq BED file from the link `https://sourceforge.net/projects/rseqc/files/BED/Mouse_Mus_musculus/mm10_RefSeq.bed.gz/download` (and rename to `reference genes`) or import a file from Shared data if provided. Alternatively, you can provide your own BED file of reference genes, for example from UCSC (see the Peaks to Genes tutorial. Or the **Convert GTF to BED12** tool can be used to convert a GTF into a BED file.

Hands-on: Check strandness with **Infer Experiment**

1. **Infer Experiment** with the following parameters:
   - *"Input .bam file"*: `aligned reads (BAM)` (output of **HISAT2**)
   - *"Reference gene model"*: `reference genes` (Reference BED file)
2. **MultiQC** with the following parameters:
   - In *"1: Results"*:
     - *"Which tool was used generate logs?"*: `RSeQC`
       - *"Type of RSeQC output?"*: `infer_experiment`
         - *"RSeQC infer_experiment output"*: `Infer Experiment output` (output of **Infer Experiment**)
3. Inspect the `Webpage` output from MultiQC

Link to here | FAQs | Gitter Chat | Help Forum

The MultiQC plot below shows the result from the full dataset for comparison.

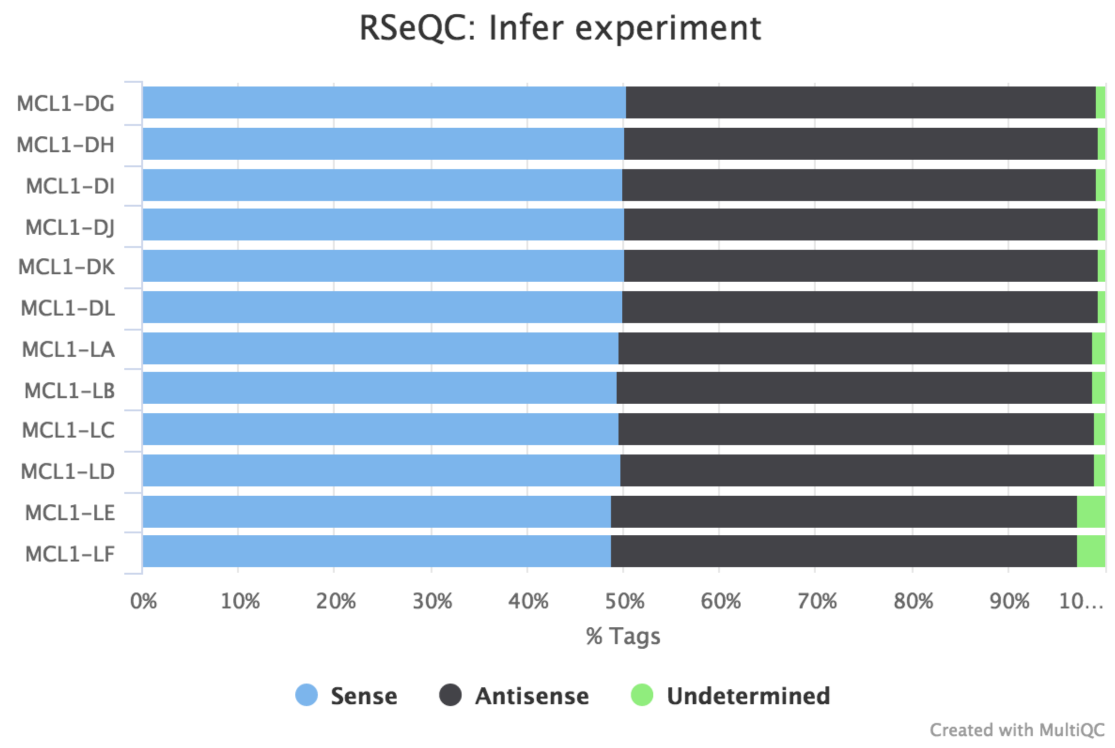


Figure 18: Infer Experiment

Question

Do you think the data is stranded or unstranded?

Solution

Duplicate reads

Duplicate reads are usually kept in RNA-seq differential expression analysis as they can come from highly-expressed genes but it is still a good metric to check. A high percentage of duplicates can indicate a problem with the sample, for example, PCR amplification of a low complexity library (not many transcripts) due to not enough RNA used as input. FastQC gives us an idea of duplicates in the reads before mapping (note that it just takes a sample of the data). We can assess the numbers of duplicates in all mapped reads using the **Picard MarkDuplicates** tool. Picard considers duplicates to be reads that map to the same location, based on the start position of where the read maps. In general, we consider normal to obtain up to 50% of duplication.

Hands-on: Check duplicate reads with **MarkDuplicates**

1. **MarkDuplicates** with the following parameters:
   - *"Select SAM/BAM dataset or dataset collection"*: `aligned reads (BAM)` (output of **HISAT2**)
2. **MultiQC** with the following parameters:
   - In *"1: Results"*:
     - *"Which tool was used generate logs?"*: `Picard`
       - *"Type of Picard output?"*: `Markdups`
         - *"Picard output"*: `MarkDuplicate metrics` (output of **MarkDuplicates**)
3. Inspect the `Webpage` output from MultiQC

Link to here | FAQs | Gitter Chat | Help Forum

The MultiQC plot below shows the result from the full dataset for comparison.

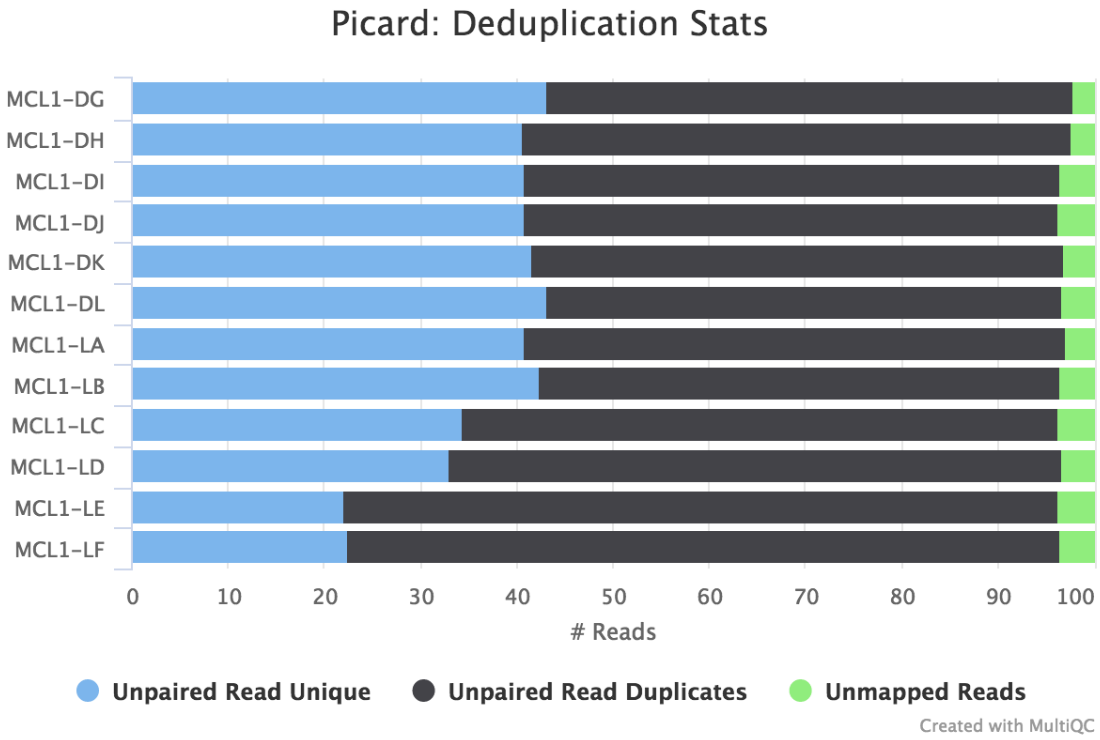


Figure 19: MarkDups metrics

Question

Which two samples have the most duplicates detected?

Solution

Reads mapped to chromosomes

You can check the numbers of reads mapped to each chromosome with the **Samtools IdxStats** tool. This can help assess the sample quality, for example, if there is an excess of mitochondrial contamination. It could also help to check the sex of the sample through the numbers of reads mapping to X/Y or to see if any chromosomes have highly expressed genes.

Hands-on: Count reads mapping to each chromosome with **IdxStats**

1. **IdxStats** with the following parameters:
   - *"BAM file"*: `aligned reads (BAM)` (output of **HISAT2**)
2. **MultiQC** with the following parameters:
   - In *"1: Results"*:
     - *"Which tool was used generate logs?"*: `Samtools`
       - *"Type of Samtools output?"*: `idxstats`
         - *"Samtools idxstats output"*: `IdxStats output` (output of **IdxStats**)
3. Inspect the `Webpage` output from MultiQC

Link to here | FAQs | Gitter Chat | Help Forum

The MultiQC plot below shows the result from the full dataset for comparison.

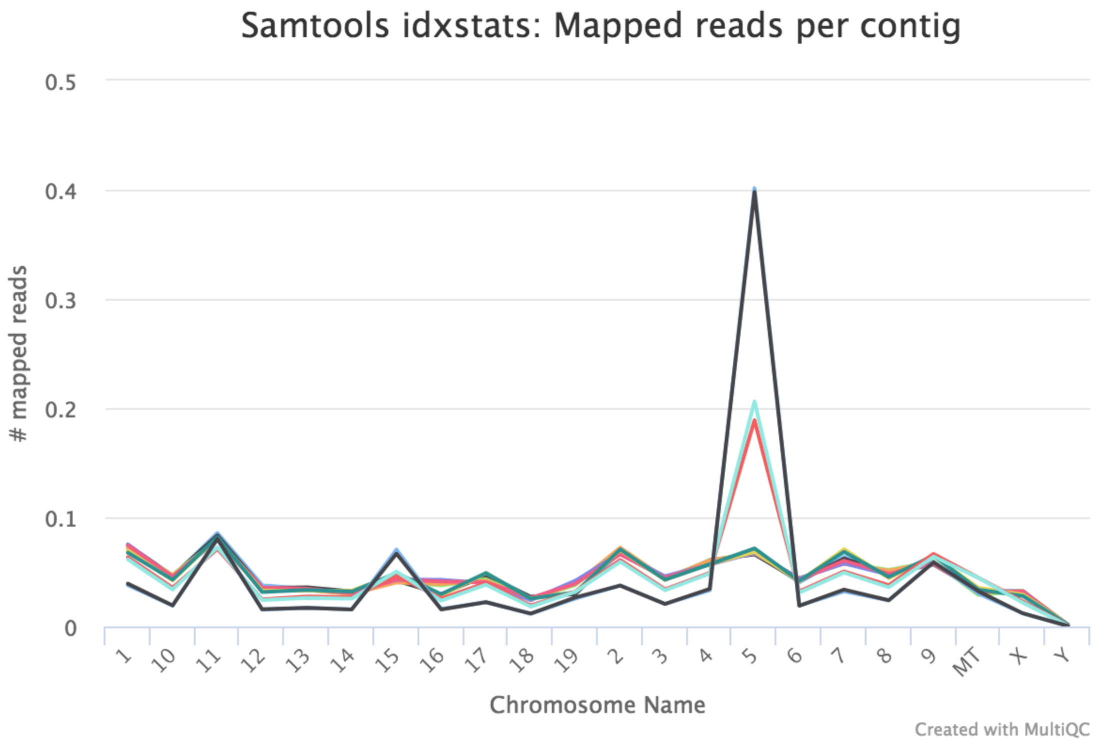


Figure 20: IdxStats Chromosome Mappings

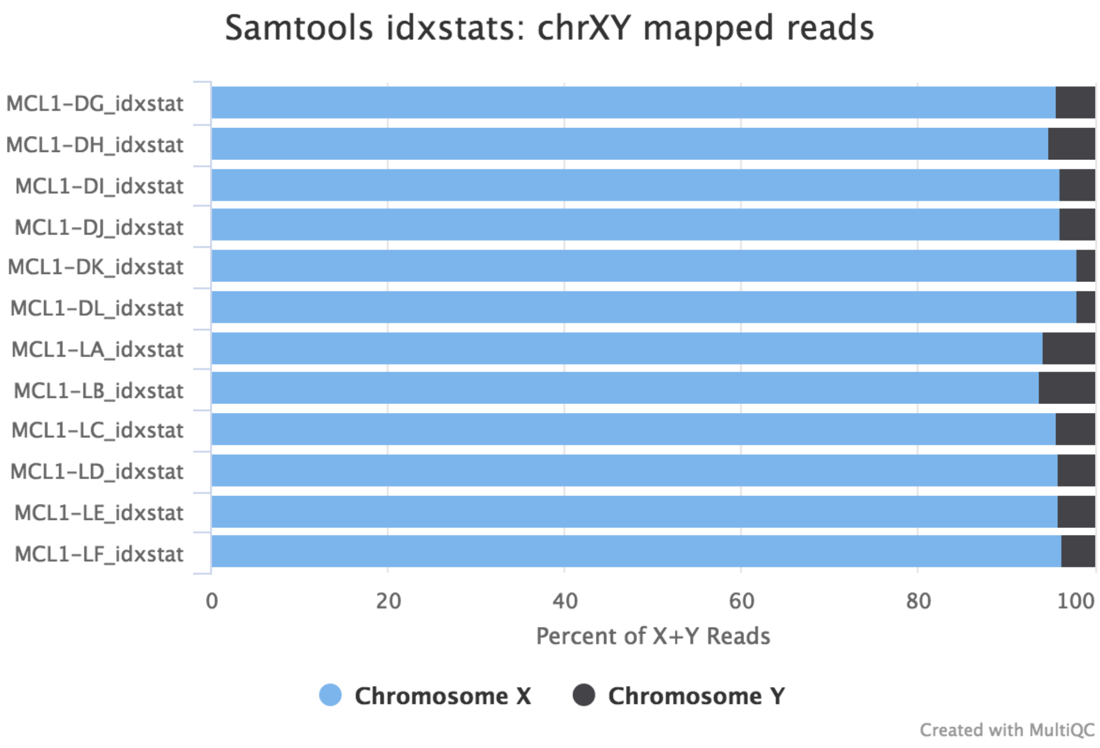


Figure 21: IdxStats X/Y Mappings

Question

1. What do you think of the chromosome mappings?
2. Are the samples male or female? *(If a sample is not in the XY plot it means no reads mapped to Y)*

Solution

Gene body coverage (5'-3')

The coverage of reads along gene bodies can be assessed to check if there is any bias in coverage. For example, a bias towards the 3' end of genes could indicate degradation of the RNA. Alternatively, a 3' bias could indicate that the data is from a 3' assay (e.g. oligodT-primed, 3'RNA-seq). You can use the RSeQC **Gene Body Coverage (BAM)** tool to assess gene body coverage in the BAM files.

Hands-on: Check coverage of genes with **Gene Body Coverage (BAM)**

1. **Gene Body Coverage (BAM)** with the following parameters:
   - *"Run each sample separately, or combine mutiple samples into one plot"*: `Run each sample separately`
     - *"Input .bam file"*: `aligned reads (BAM)` (output of **HISAT2**)
   - *"Reference gene model"*: `reference genes` (Input dataset)
2. **MultiQC** with the following parameters:
   - In *"1: Results"*:
     - *"Which tool was used generate logs?"*: `RSeQC`
       - *"Type of RSeQC output?"*: `gene_body_coverage`
         - *"RSeQC gene_body_coverage output"*: `Gene Body Coverage (BAM) (text)` (output of **Gene Body Coverage (BAM)**)
3. Inspect the `Webpage` output from MultiQC

Link to here | FAQs | Gitter Chat | Help Forum

The MultiQC plot below shows the result from the full dataset for comparison.

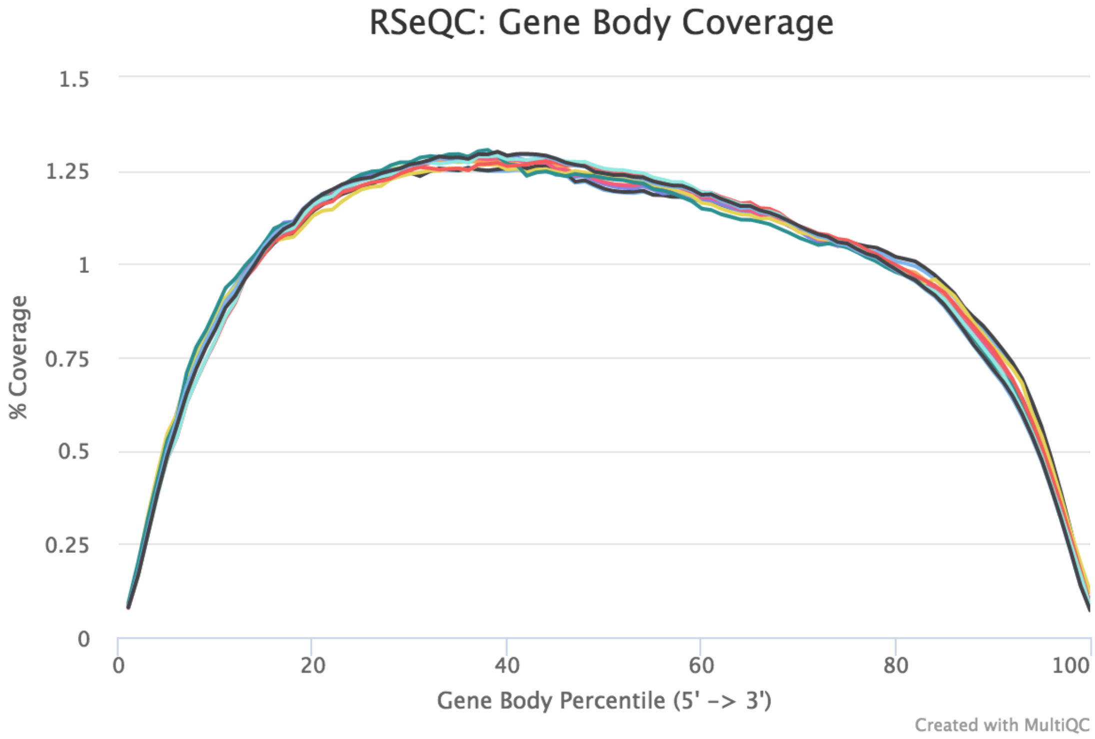


Figure 22: Gene Body Coverage

The plot below from the RSeQC website shows what samples with 3'biased coverage would look like.

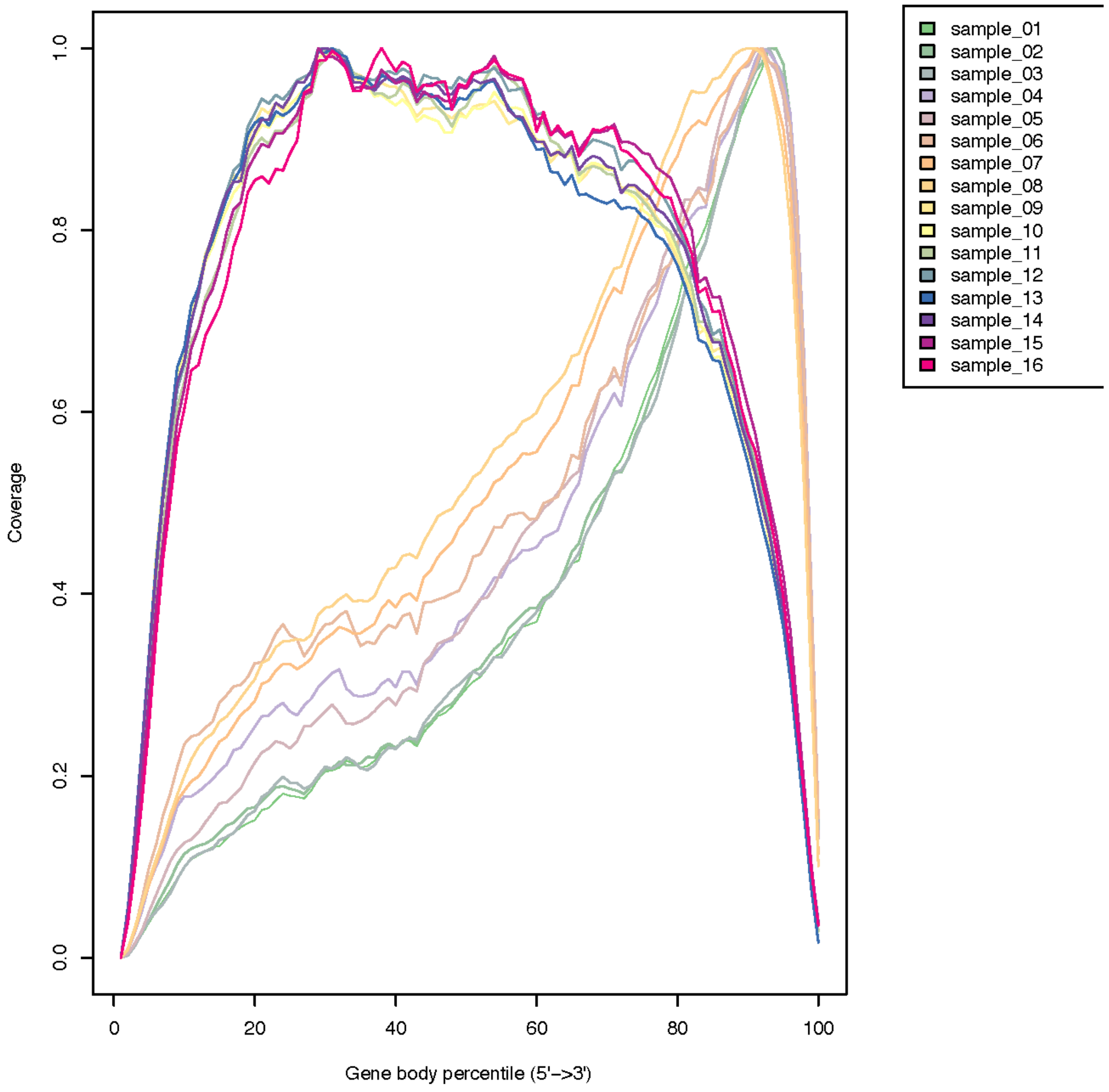


Figure 23: Gene Body Coverage comparison

Question

What do you think of the coverage across gene bodies in these samples?

Solution

Read distribution across features (exons, introns, intergenic..)

We can also check the distribution of reads across known gene features, such as exons (CDS, 5’UTR, 3’UTR), introns and intergenic regions. In RNA-seq we expect most reads to map to exons rather than introns or intergenic regions. It is also the reads mapped to exons that will be counted so it is good to check what proportions of reads have mapped to those. High numbers of reads mapping to intergenic regions could indicate the presence of DNA contamination.

Hands-on: Check distribution of reads with **Read Distribution**

1. **Read Distribution** with the following parameters:
   - *"Input .bam/.sam file"*: `aligned reads (BAM)` (output of **HISAT2**)
   - *"Reference gene model"*: `reference genes` (Input dataset)
2. **MultiQC** with the following parameters:
   - In *"1: Results"*:
     - *"Which tool was used generate logs?"*: `RSeQC`
       - *"Type of RSeQC output?"*: `read_distribution`
         - *"RSeQC read_distribution output"*: `Read Distribution output` (output of **Read Distribution**)
3. Inspect the `Webpage` output from MultiQC

Link to here | FAQs | Gitter Chat | Help Forum

The MultiQC plot below shows the result from the full dataset for comparison.

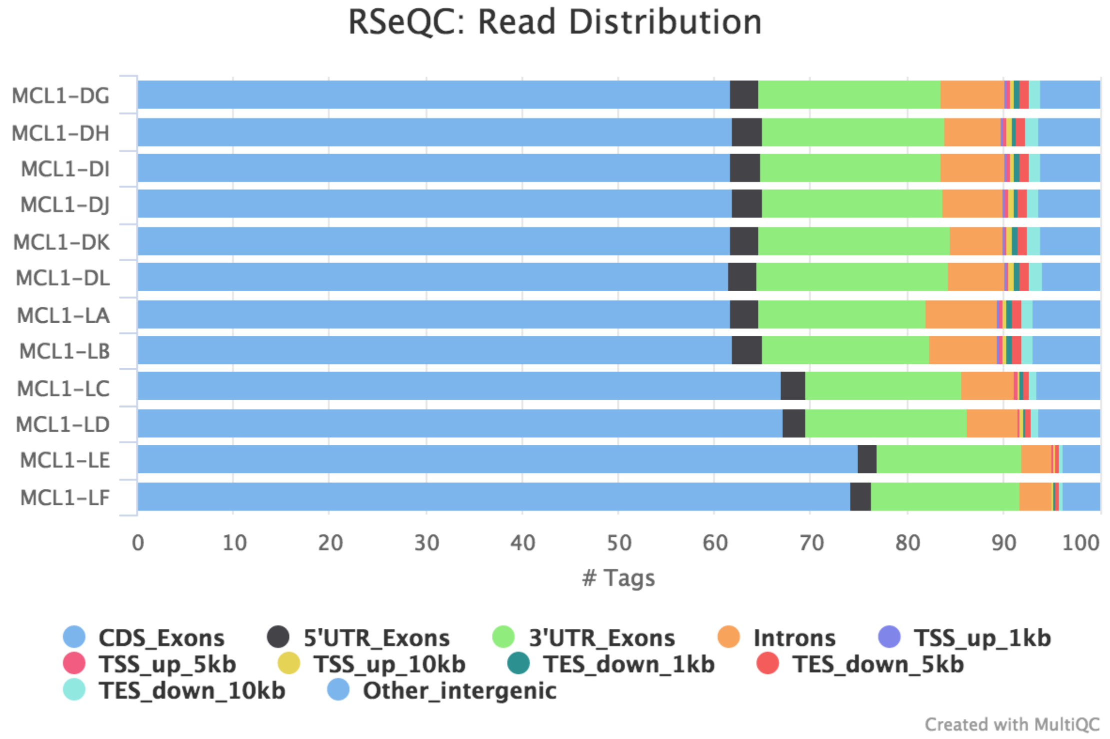


Figure 24: Read Distribution

Question

What do you think of the read distribution?

👁 Solution ➕

The MultiQC report can be downloaded by clicking on the floppy disk icon on the dataset in the history.

? Question

Can you think of any other QCs that could be performed on RNA-seq reads?

👁 Solution ➕

Conclusion

In this tutorial we have seen how reads (FASTQ files) can be converted into counts. We have also seen QC steps that can be performed to help assess the quality of the data. A follow-on tutorial, RNA-seq counts to genes, shows how to perform differential expression and QC on the counts for this dataset.

🔑 Key points

- In RNA-seq, reads (FASTQs) are mapped to a reference genome with a spliced aligner (e.g HISAT2, STAR)
- The aligned reads (BAMs) can then be converted to counts
- Many QC steps can be performed to help check the quality of the data
- MultiQC can be used to create a nice summary report of QC information
- The Galaxy Rule-based Uploader, Collections and Workflows can help make analysis more efficient and easier

Frequently Asked Questions

Have questions about this tutorial? Check out the tutorial FAQ page or the FAQ page for the Transcriptomics topic to see if your question is listed there. If not, please ask your question on the GTN Gitter Channel or the Galaxy Help Forum

Useful literature

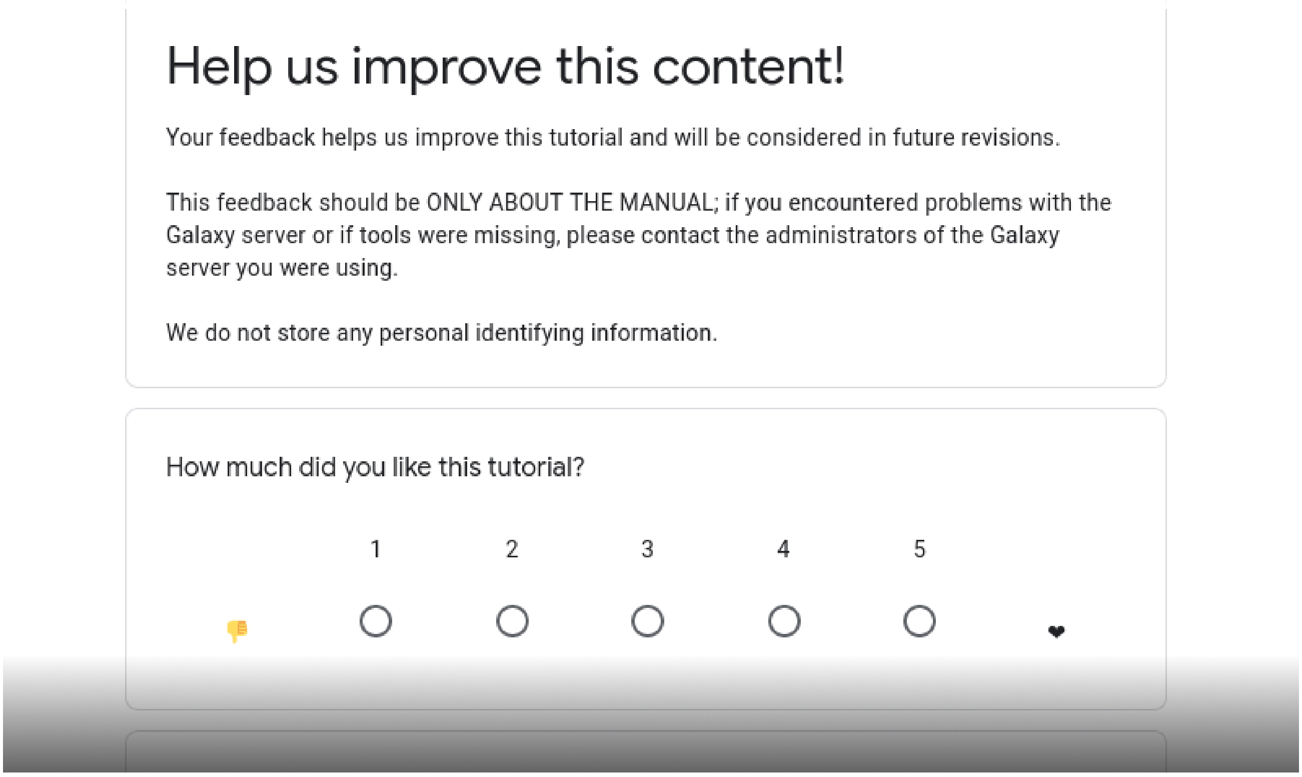
Further information, including links to documentation and original publications, regarding the tools, analysis techniques and the interpretation of results described in this tutorial can be found here.

References

1. Marcel, M., 2011 **Cutadapt removes adapter sequences from high-throughput sequencing reads**. EMBnet.journal 17: 10–12. http://journal.embnet.org/index.php/embnetjournal/article/view/200
2. Wang, L., S. Wang, and W. Li, 2012 **RSeQC: quality control of RNA-seq experiments**. Bioinformatics 28: 2184–2185. https://www.ncbi.nlm.nih.gov/pubmed/22743226
3. Dobin, A., C. A. Davis, F. Schlesinger, J. Drenkow, C. Zaleski *et al.*, 2013 **STAR: ultrafast universal RNA-seq aligner**. Bioinformatics 29: 15–21. https://academic.oup.com/bioinformatics/article/29/1/15/272537
4. Liao, Y., G. K. Smyth, and W. Shi, 2013 **featureCounts: an efficient general purpose program for assigning sequence reads to genomic features**. Bioinformatics 30: 923–930. https://academic.oup.com/bioinformatics/article/30/7/923/232889
5. Liao, Y., G. K. Smyth, and W. Shi, 2013 **The Subread aligner: fast, accurate and scalable read mapping by seed-and-vote**. Nucleic Acids Research 41: e108–e108. 10.1093/nar/gkt214
6. Fu, N. Y., A. C. Rios, B. Pal, R. Soetanto, A. T. L. Lun *et al.*, 2015 **EGF-mediated induction of Mcl-1 at the switch to lactation is essential for alveolar cell survival**. Nature Cell Biology 17: 365–375. 10.1038/ncb3117
7. Kim, D., B. Langmead, and S. L. Salzberg, 2015 **HISAT: a fast spliced aligner with low memory requirements**. Nature Methods 12: 357. https://www.nature.com/articles/nmeth.3317
8. Ewels, P., M. Magnusson, S. Lundin, and M. Käller, 2016 **MultiQC: summarize analysis results for multiple tools and samples in a single report**. Bioinformatics 32: 3047–3048. https://academic.oup.com/bioinformatics/article/32/19/3047/2196507

Feedback

Did you use this material as an instructor? Feel free to give us feedback on how it went.
Did you use this material as a learner or student? Click the form below to leave feedback.👇



Help us improve this content!

Your feedback helps us improve this tutorial and will be considered in future revisions.

This feedback should be ONLY ABOUT THE MANUAL; if you encountered problems with the Galaxy server or if tools were missing, please contact the administrators of the Galaxy server you were using.

We do not store any personal identifying information.

How much did you like this tutorial?

	1	2	3	4	5	
👎	○	○	○	○	○	❤

Citing this Tutorial

1. Maria Doyle, Belinda Phipson, Harriet Dashnow, **1: RNA-Seq reads to counts (Galaxy Training Materials)**. https://training.galaxyproject.org/training-material/topics/transcriptomics/tutorials/rna-seq-reads-to-counts/tutorial.html Online; accessed TODAY
2. Batut et al., 2018 **Community-Driven Data Analysis Training for Biology** Cell Systems 10.1016/j.cels.2018.05.012

BibTeX

Congratulations on successfully completing this tutorial!

Do you want to extend your knowledge? Follow one of our recommended follow-up trainings:

- Transcriptomics
  - 2: RNA-seq counts to genes: hands-on