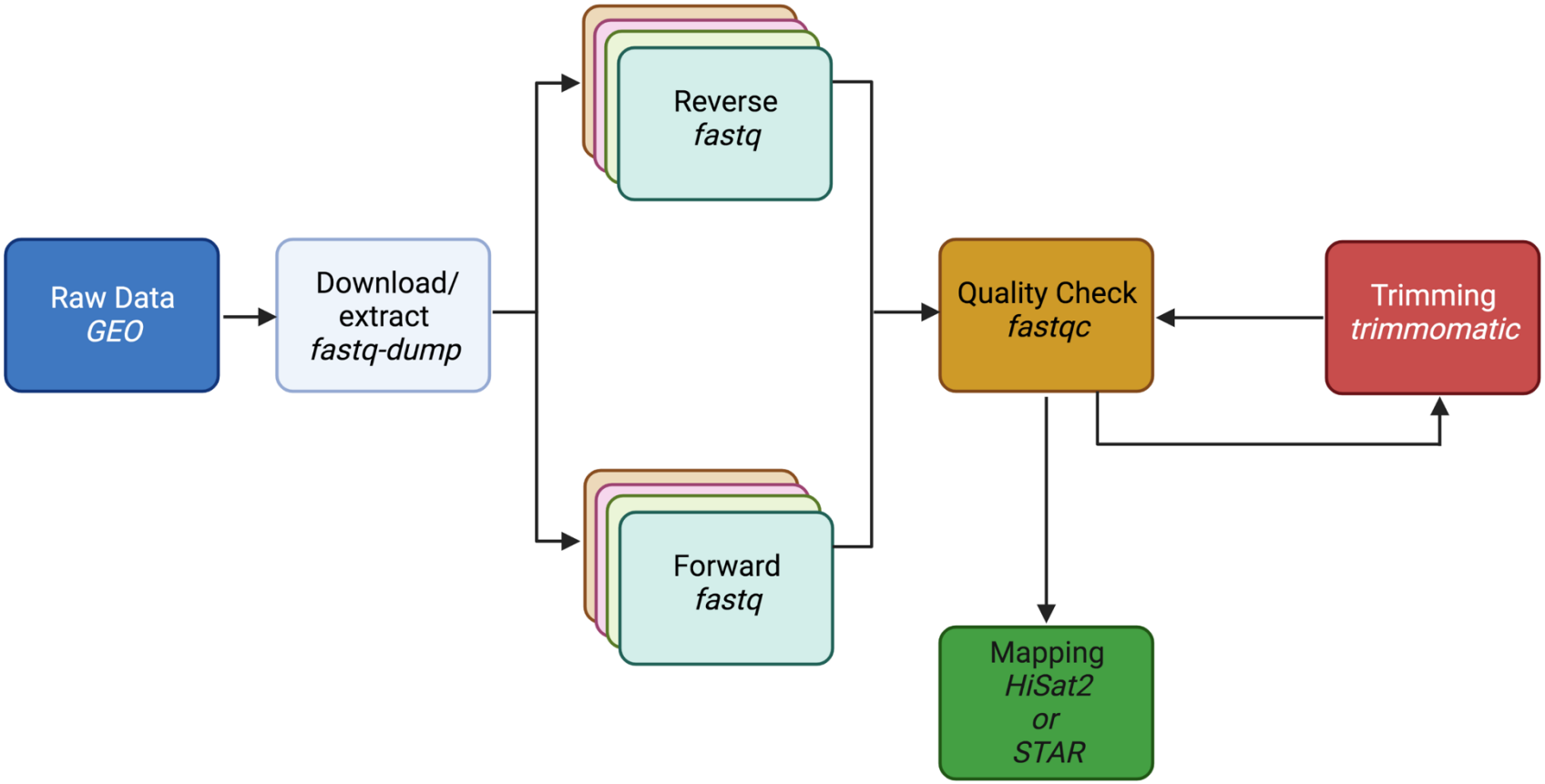
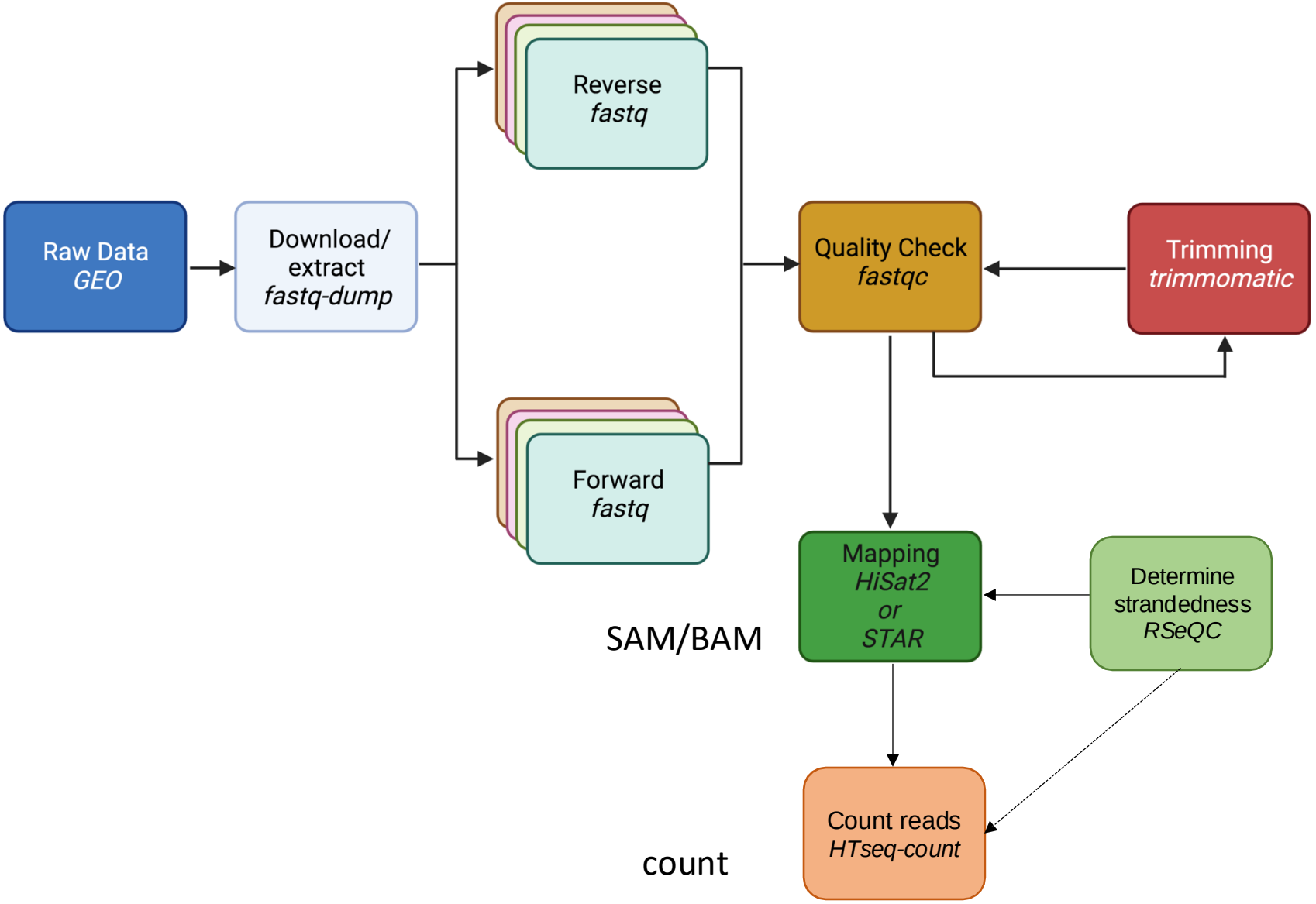
RSeQC & HTSeq

March 3rd, 2025


Raw Data
GEO
Download/
extract
fastq-dump
Reverse
fastq
Forward
fastq
Quality Check
fastqc
Trimming
trimmomatic
Mapping
HiSat2
or
STAR


Raw Data
GEO
Download/
extract
fastq-dump
Reverse
fastq
Forward
fastq
Quality Check
fastqc
Trimming
trimmomatic
Mapping
HiSat2
or
STAR
SAM/BAM
Determine
strandedness
RSeQC
Count reads
HTseq-count
count

Pre & post alignment QC

- **raw reads** QC
  - adapter/primer/other contaminating and over-represented sequences
  - sequencing quality
  - GC distributions
  - duplication levels

- **aligned reads** QC
  - % (uniquely) aligned reads
  - % exonic vs. intronic/intergenic
  - strandedness
  - splice junction detection/annotation

Pre-alignment:
FastQC, fastp

Post-alignment:
RSeQC, QoRTs

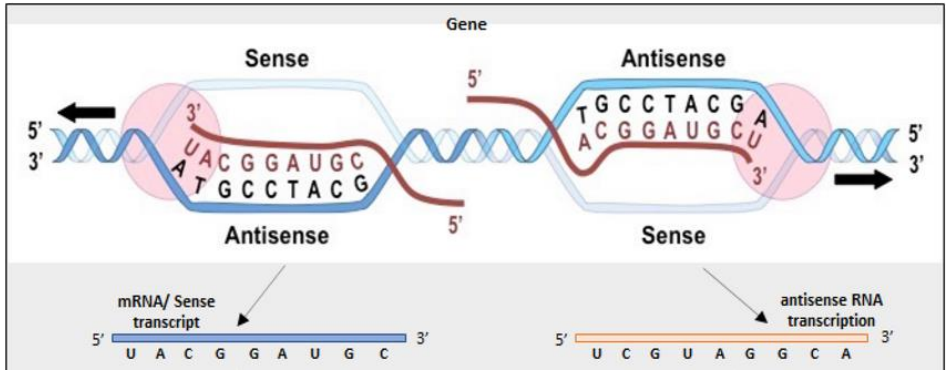
Stranded libraries

A major decision to be made during the library preparation step is whether to preserve **RNA strand information**.

Unlike DNA molecules, RNA molecules exist as single-stranded threads that could result from the **sense** or **antisense strand**.

The creation of stranded libraries are now standard with Illumina TruSeq 'stranded' RNA-Seq kits

Allows for the identification of which strand of DNA the RNA was transcribed from



RNA-Seq library preparation protocols

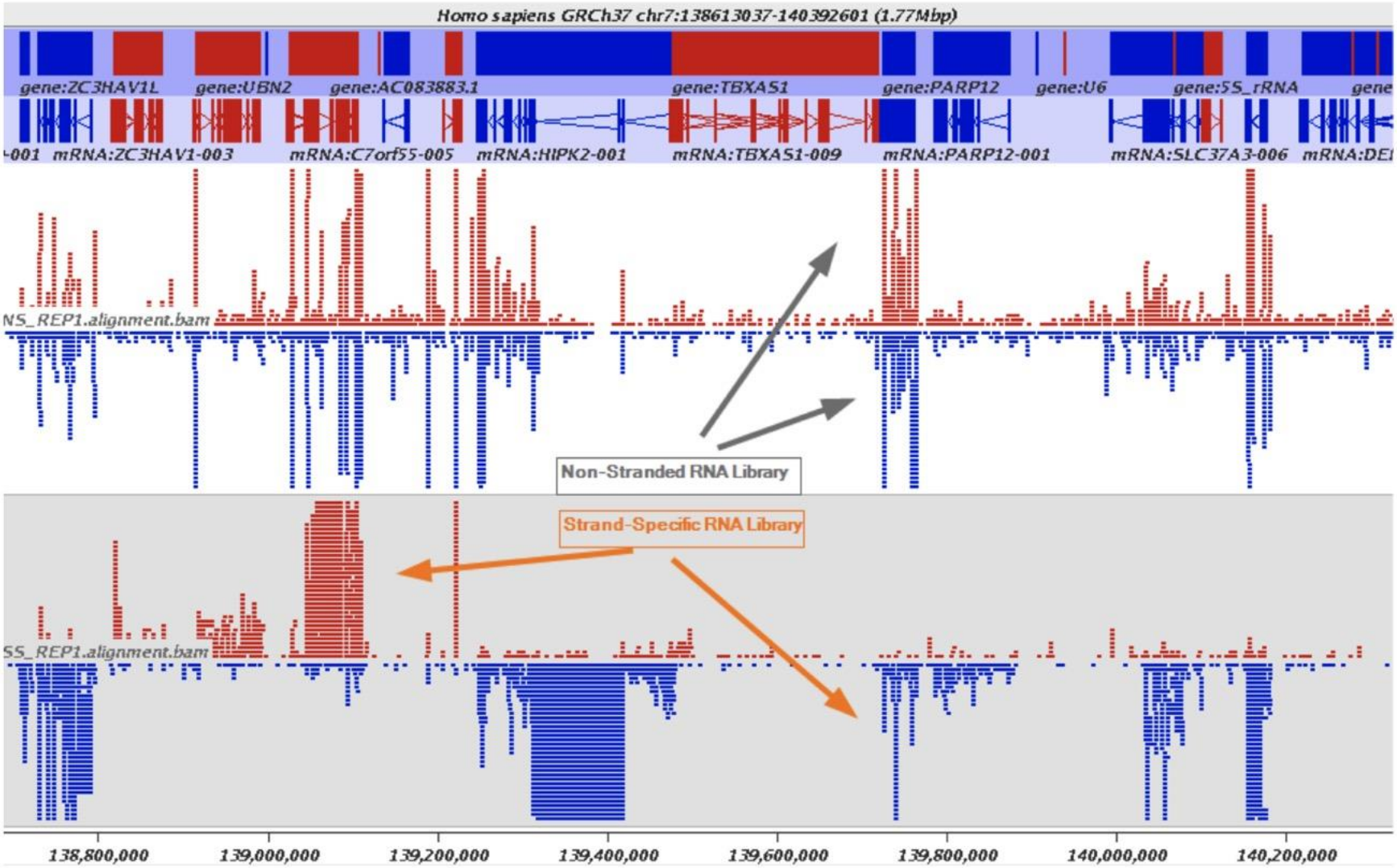
Strandedness	Example Protocols
Unstranded	Standard Illumina mRNA-Seq
Forward-stranded (1st strand)	Illumina TruSeq Stranded, NEBNext Ultra Directional RNA
Reverse-stranded (2nd strand)	Illumina ScriptSeq, SMARTer Stranded

After today's activity you should be modifying your hisat2_final_proj.sh script!

Why retain stranded information?

- It makes sense to begin with the most information possible – even if immediately that is not of interest
- Useful for identifying antisense transcripts, mapping splicing events, and detecting overlapping transcripts.
- They are commonly used in studies of transcriptomics, gene expression analysis, and RNA editing, and *de novo* assembly.

The implication of stranded RNAseq is that you can distinguish whether the reads are derived from forward- or reverse-encoded transcripts:



Stranded RNAseq data look like this

This example contrasts unstranded and stranded RNAseq experiments. Red transcripts are from + strand and blue are from - strand. In stranded example reads are clearly stratified between the two strands. A small number of reads from opposite strand may represent anti-sense transcription. The image from GATC Biotech.

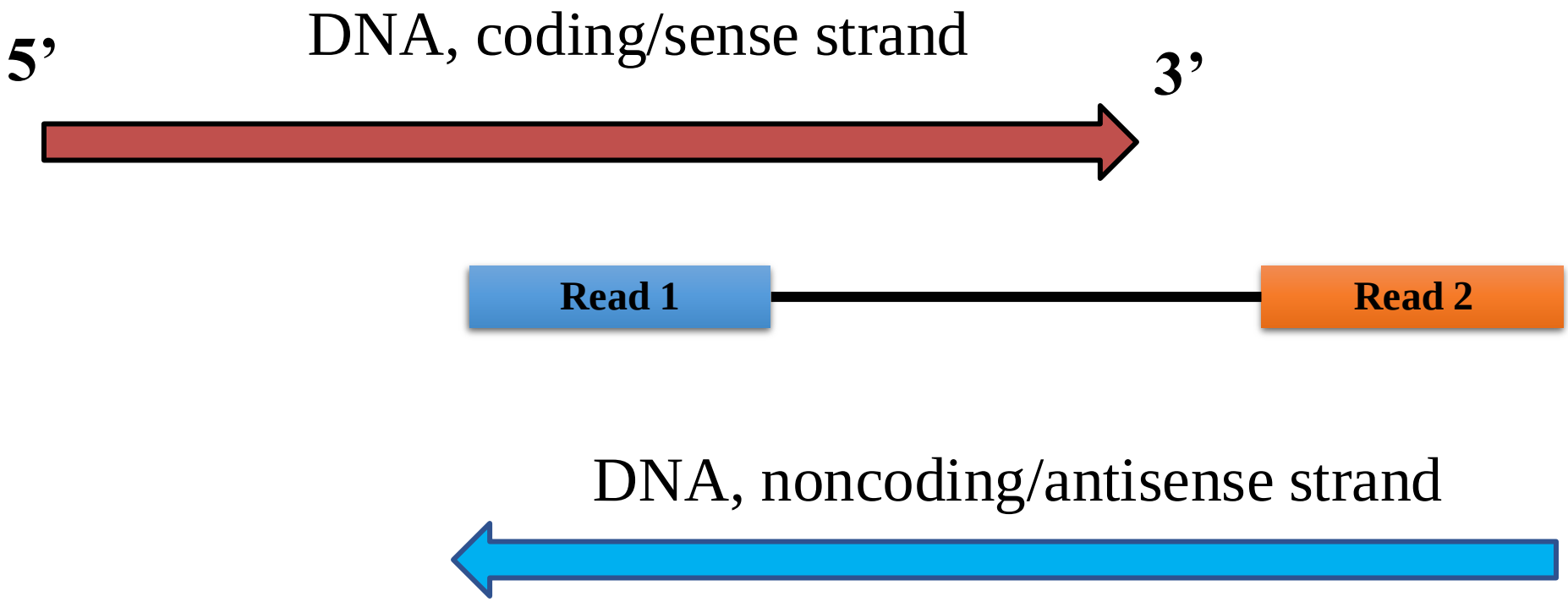
Why is strandedness important to determine?

- If you use wrong directionality parameter in the alignment or read counting step, the reads may orient to the *wrong strand*.
- This means you may not get any read counts or if there is a gene in the same location these reads can be counted for the wrong gene.
- Its important to check using QC tools!

Three scenarios when it comes to stranded libraries

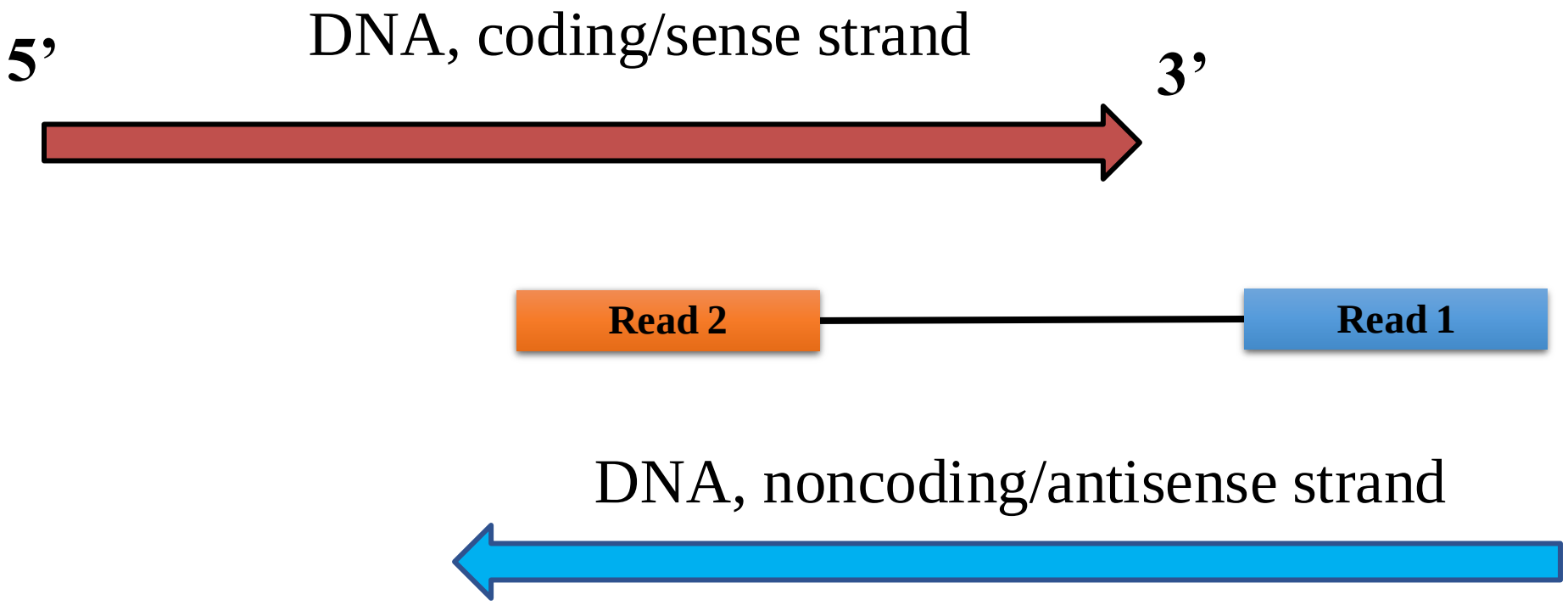
- Forward (sense) – reads resemble the DNA sequence
- Reverse (antisense) – reads resemble the complementary sequence
- Unstranded

Forward (sense)

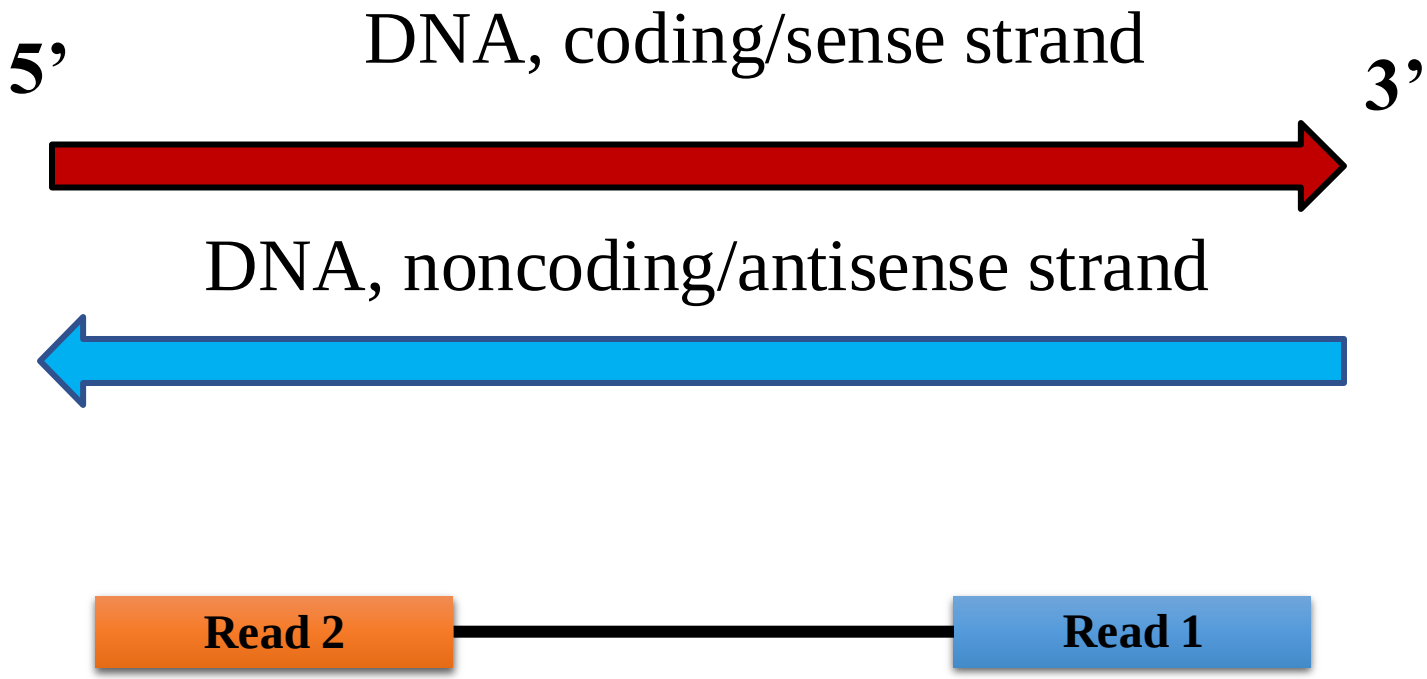


If sequences of Read 1 align to the coding, sense strand – the library is “stranded”

Reverse (antisense)



If sequences of Read 2 align to the coding, sense strand – the library is “ reverse stranded”



If both Read 1 and Read 2 align to the coding, sense strand – the library is “unstranded”

Options to select from when aligning and counting

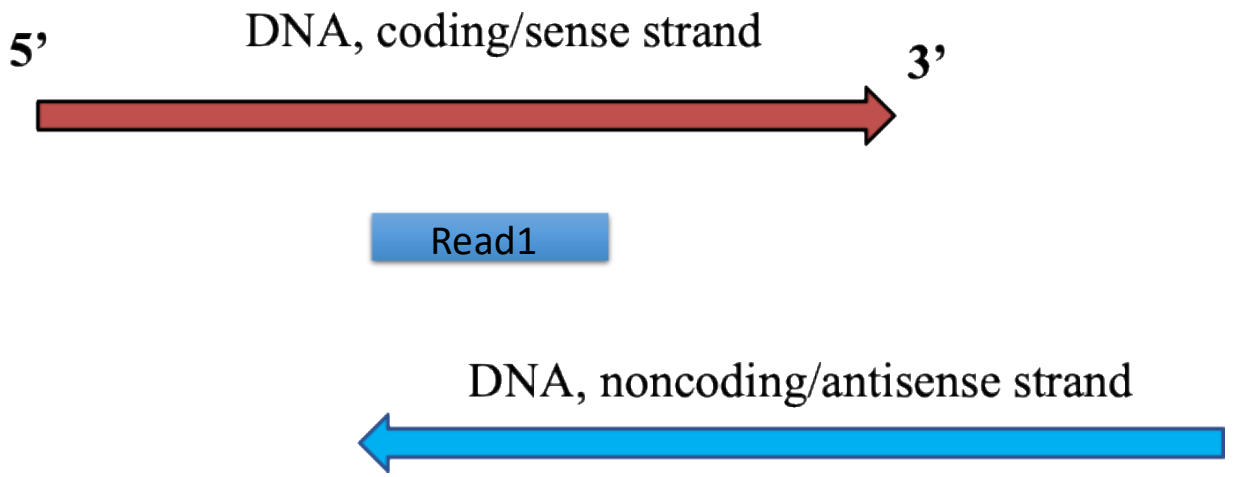
	Option 1 **Stranded/sense**	**Option 2** **Reverse/antisense**	**Option 3** **Unstranded**
HISAT2 **(**--rna-strandedness)	FR (for PE) F (for SE)	RF (for PE) R (for SE)	Default
STAR	n/a	n/a	n/a
HTSeq	stranded=yes	stranded=reverse	stranded=no

An Example HISAT2 command with *strandedness* accounted for

```
hisat2 -x genome_index -1 reads_R1.fq -2 reads_R2.fq --rna-strandedness RF -S output.sam
```

In this example, specified --rna-strandedness RF for reverse stranded libraries

However, in the context of **single-end RNA-Seq**, we only have Read 1 (R1), so it is based on whether R1 maps to the **sense or antisense strand**.



Today's Exercises

- All will utilize a "package of python scripts" bundled as RSeQC

infer_experiment.py
bam_stat.py
junction_saturation.py
junction_annotation.py
read_distribution.py

Strand-Specificity

infer_experiment.py

Why use it?

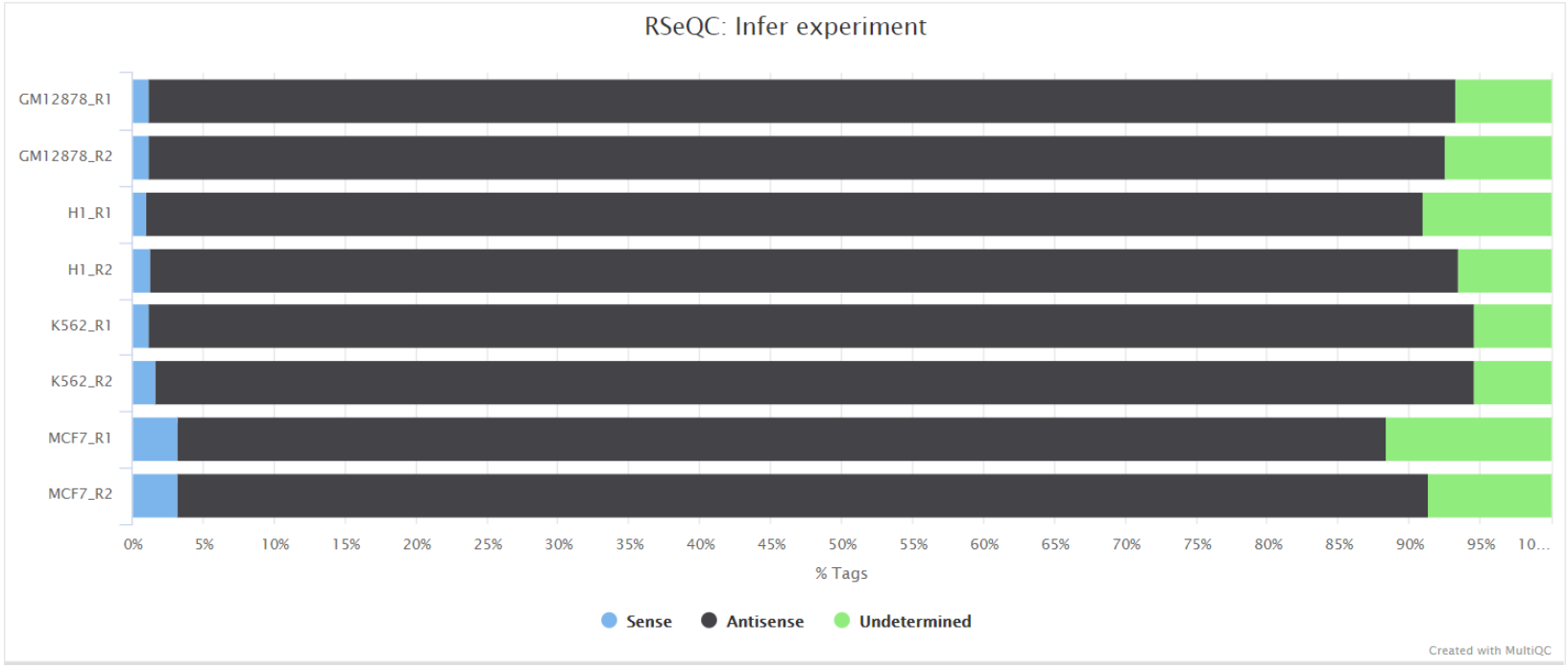
- Crucial for differential expression analysis as strand specific libraries require correct orientation for accurate read assignment
- Avoids misinterpretation of gene expression levels causes by incorrect strand assignment

Strand-Specificity

infer_experiment.py

MultiQC

Expected Output:



Class Exercise #1

- Perform steps under the Getting Started Section first
- Our goal is to understand which option to select for Irrel_kd_1.subset_sorted.bam.

	Option 1 **Stranded/sense**	**Option 2** **Reverse/antisense**	**Option 3** **Unstranded**
HISAT2 **(**--rna-strandedness)	FR (for PE) F (for SE)	RF (for PE) R (for SE)	Default
STAR	n/a	n/a	n/a
HTSeq	stranded=yes	stranded=reverse	stranded=no

Class Exercise #1

- Before getting started:
  - Read Planning and Organization
  - And perform steps under “Getting Started Section”

- If done correctly, look inside of **Irrel_kd_1.subset_read.log:**

Reading reference gene model ../../RSeQC_bed_files/refseq.hg38.bed12 ... Done
Loading SAM/BAM file ... Finished
Total 193075 usable reads were sampled

This is SingleEnd Data
Fraction of reads failed to determine: 0.0920
Fraction of reads explained by "++,--": 0.0168
Fraction of reads explained by "+-,-+": 0.8912

BAM Statistics

bam_stat.py

Why use it?

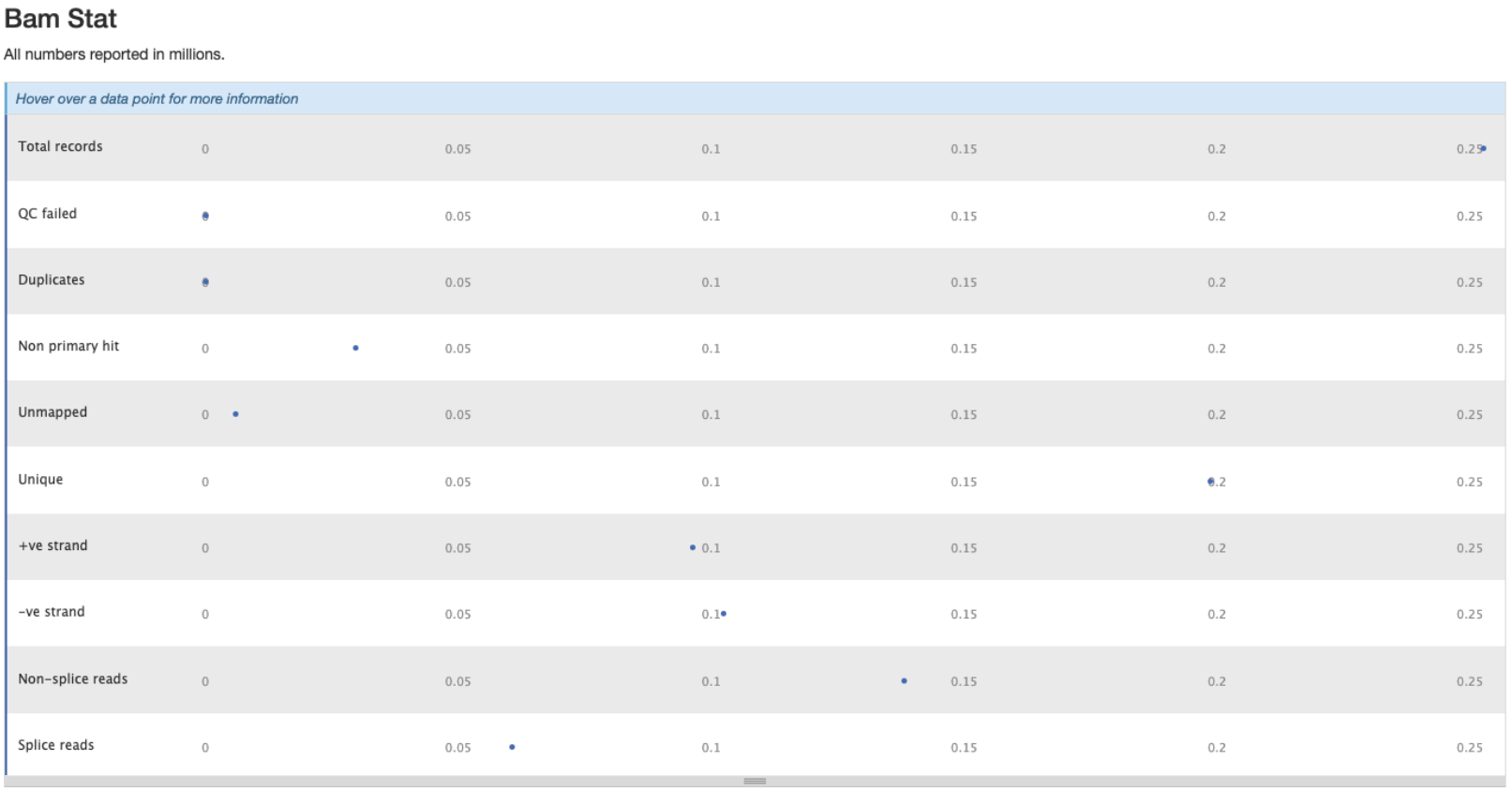
- Quickly checks overall alignment quality
- Reports metrics such as total reads, mapped reads

Strand-Specificity

bam_stat.py

MultiQC

Expected Output:



Class Exercise #2

- Our goal is to understand do we see a high (<90%) mapping rate for Irrel_kd_1.subset_sorted.bam?

- If done correctly, look inside of **Irrel_kd_1.subset_bamstat.log:**

Load BAM file ... Done
#==========#
All numbers are READ count
#==========
Total records: 252558
QC failed: 0
Optical/PCR duplicate: 0
Non primary hits 29637
Unmapped reads: 5879
mapq < mapq_cut (non-unique): 18323
mapq >= mapq_cut (unique): 198719
Read-1: 0
Read-2: 0

Splice Junction Detection

junction_saturation.py

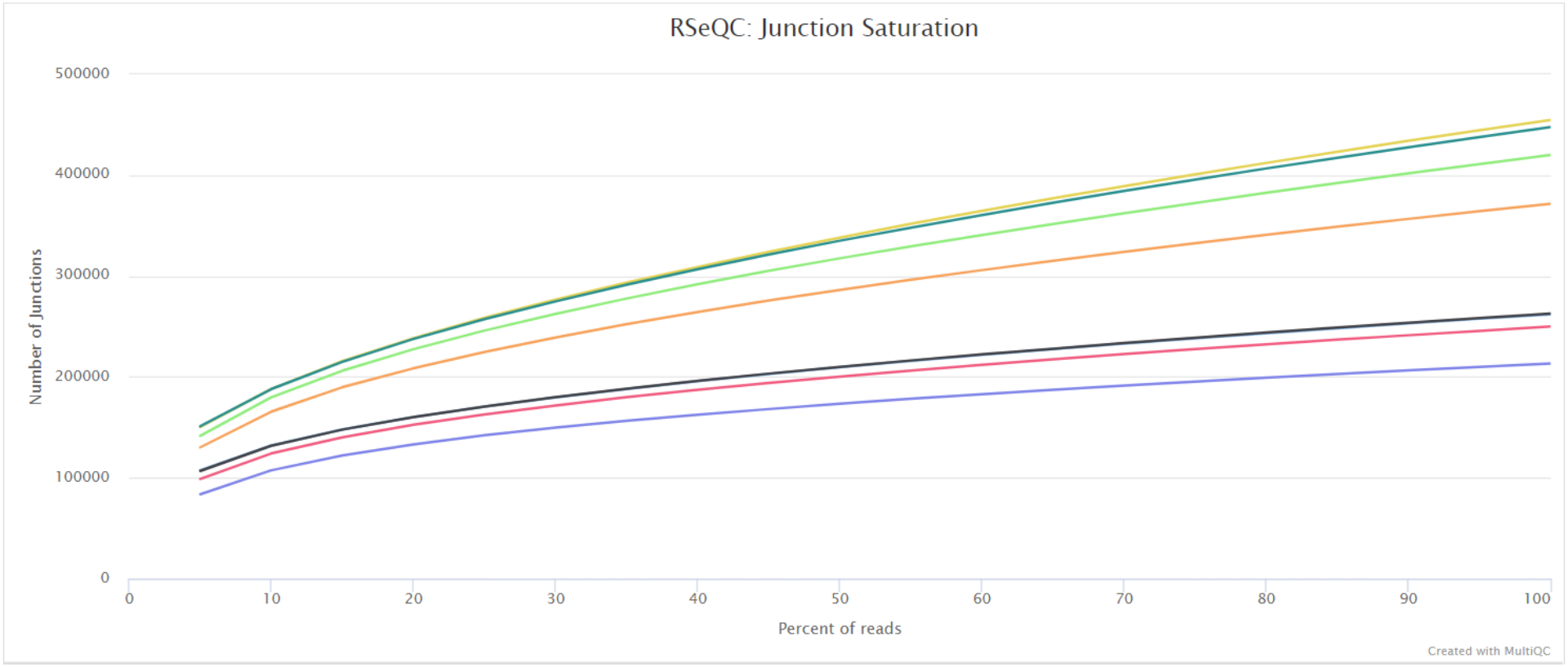
Why use it?

- Aids in determining if additional sequencing would improve splice junction discovery
- Does this dataset have sufficient coverage to detect splice junctions reliably?

Splice Junction Detection

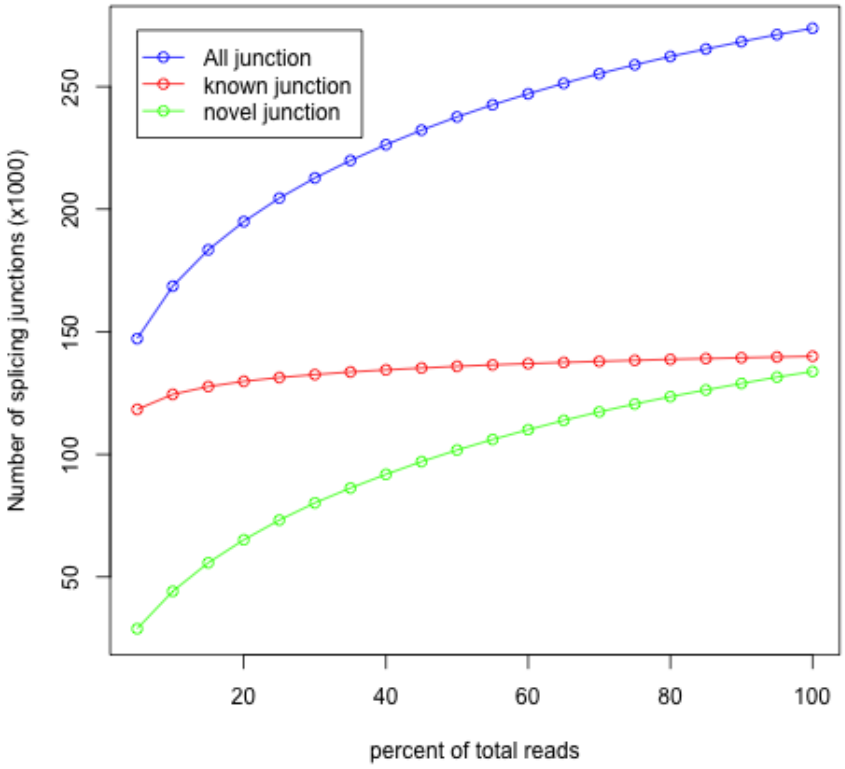
junction_saturation.py

Expected Output:

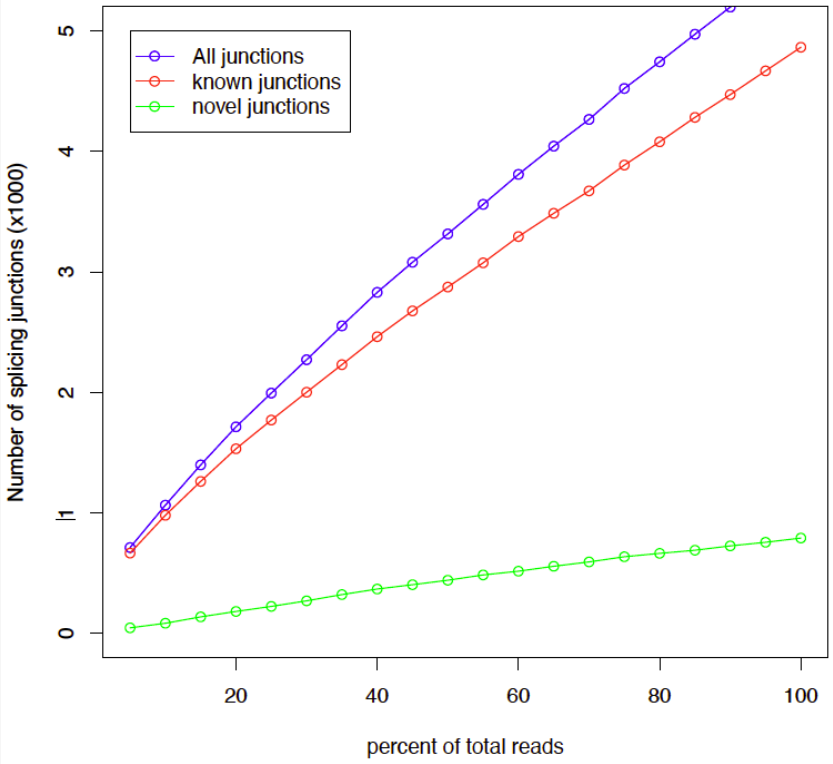


Class Exercise #3

junction_saturation.py



PDF displaying desired output
By the time it reaches, 100% see a plateau
No more reads are required = good ☺



PDF displaying undesired output
Have not reached the maximum
More reads are required to detect splice junctions = bad ☹

Splice Junction Annotation

junction_annotation.py

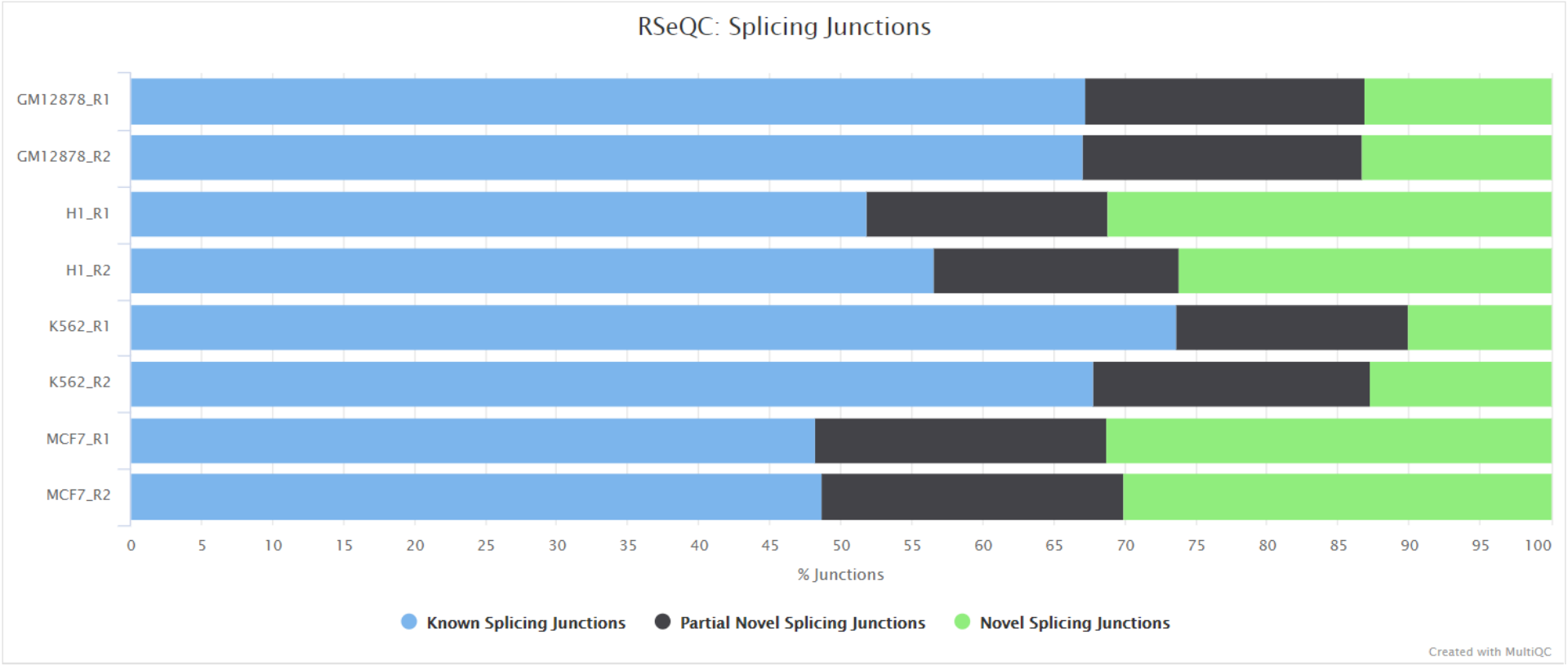
Why use it?

- Validates detected splice sites against known annotations
- Detects novel splice junctions

Splice Junction Annotation

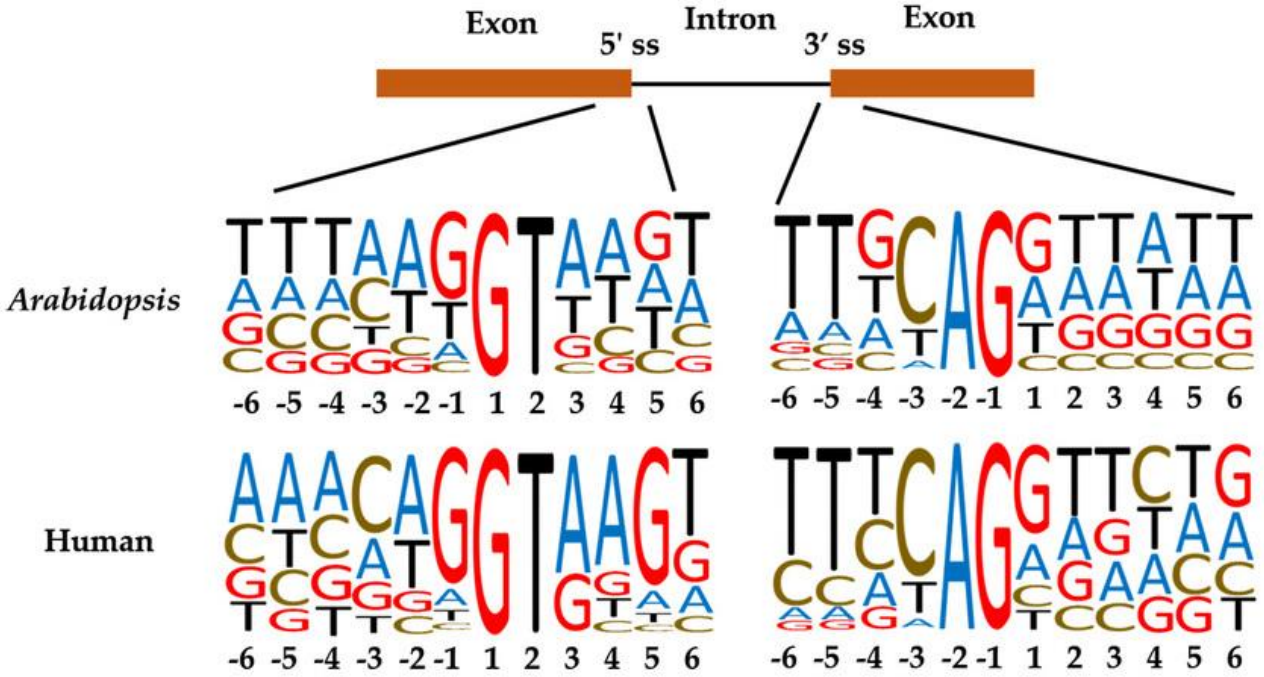
junction_annotation.py

Expected Output:



Detected junctions were divided to 3 exclusive categories:

1. Annotated (known): The junction is part of the gene model. Both splice sites, 5’ splice site (5’SS) and 3’splice site (3’SS) are annotated by reference gene model.
2. Complete_novel: Both 5’SS and 3’SS are novel.
3. Partial_novel: One of the splice site (5’SS or 3’SS) is novel, and the other splice site is annotated



Class Exercise #4

junction_annotation.py

Junction annotation

▼ Output files

- `<ALIGNER>/rseqc/junction_annotation/bed/`
  - `*.junction.bed` : BED file containing splice junctions.
  - `*.junction.Interact.bed` : BED file containing interacting splice junctions.
- `<ALIGNER>/rseqc/junction_annotation/log/`
  - `*.junction_annotation.log` : Log file generated by the program.
- `<ALIGNER>/rseqc/junction_annotation/pdf/`
  - `*.splice_events.pdf` : PDF file containing splicing events plot.
  - `*.splice_junction.pdf` : PDF file containing splice junctions plot.
- `<ALIGNER>/rseqc/junction_annotation/rscript/`
  - `*.junction_plot.r` : R script used to generate pdf plots above.
- `<ALIGNER>/rseqc/junction_annotation/xls/`
  - `*.junction.xls` : Excel spreadsheet with junction information.

Read Distribution Across Genomic Features

read_distribution.py

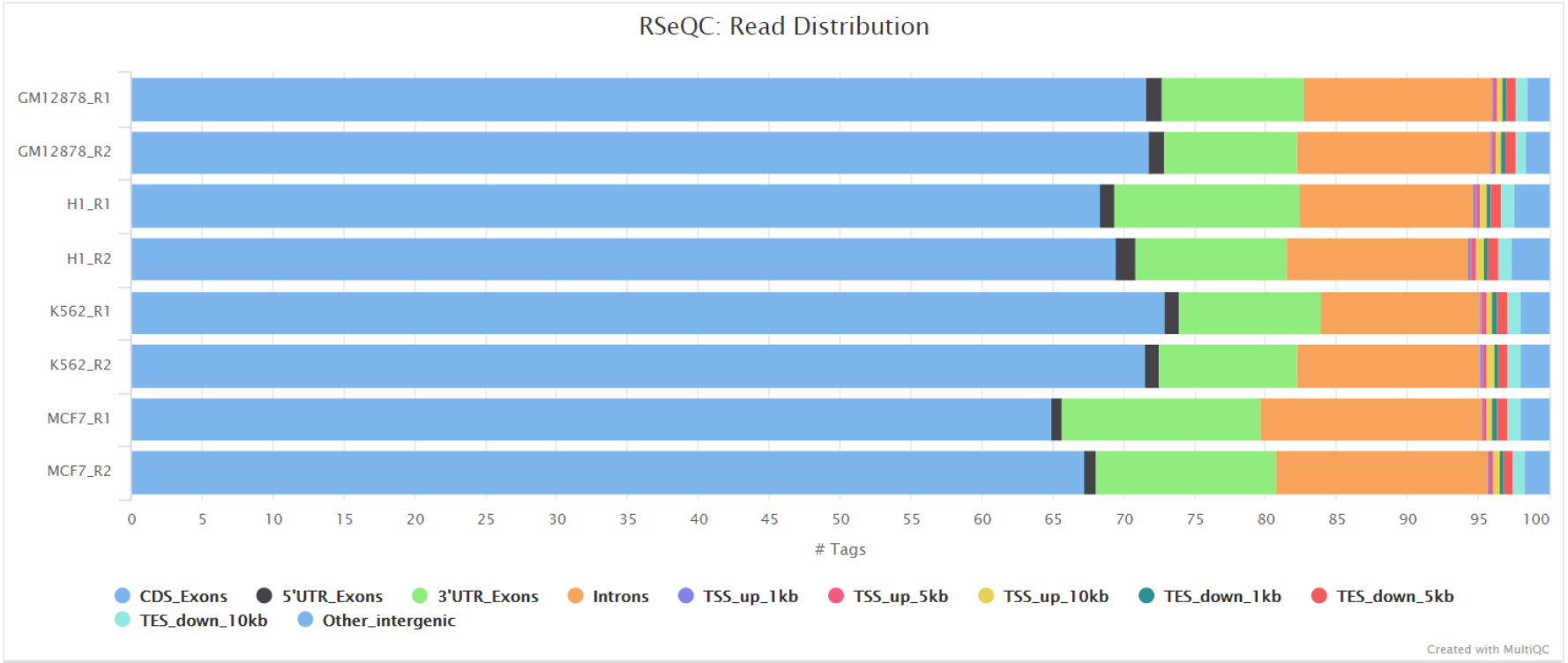
Why use it?

- Ensures the expected proportion of reads fall within exons for RNA-Seq experiments

Read Distribution Across Genomic Features

read_distribution.py

Expected Output:



For RNA-Seq, most reads (>70%) should align to exons.

Class Exercise #5

read_distribution.py

- If done correctly, look inside of **Irrel_kd_1.subset_read.log:**

Total Reads 217042
Total Tags 285170
Total Assigned Tags 270555
==========
Group Total_bases Tag_count Tags/Kb
CDS_Exons 41135377 167362 4.07
5'UTR_Exons 50276940 9969 0.20
3'UTR_Exons 74969271 67281 0.90
Introns 1591346634 21456 0.01
TSS_up_1kb 26447701 226 0.01
TSS_up_5kb 118706417 596 0.01
TSS_up_10kb 217519222 771 0.00
TES_down_1kb 30123959 846 0.03
TES_down_5kb 130727548 2947 0.02
TES_down_10kb 231251776 3716 0.02
==========