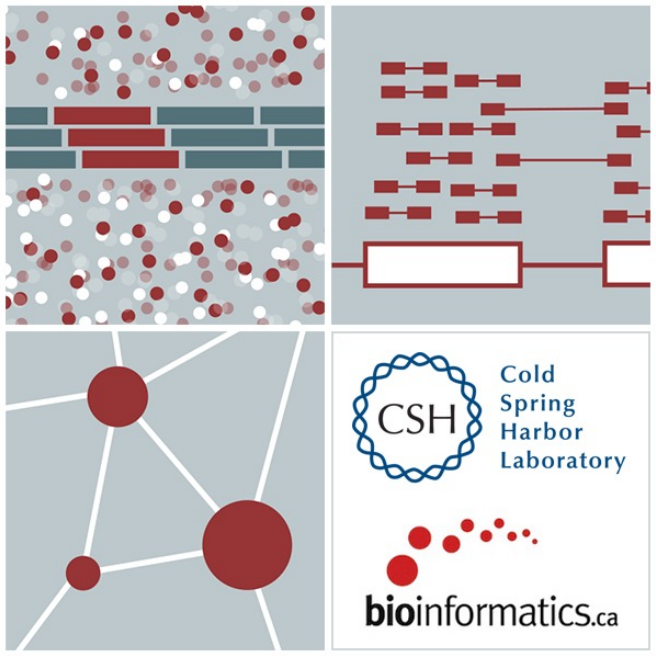
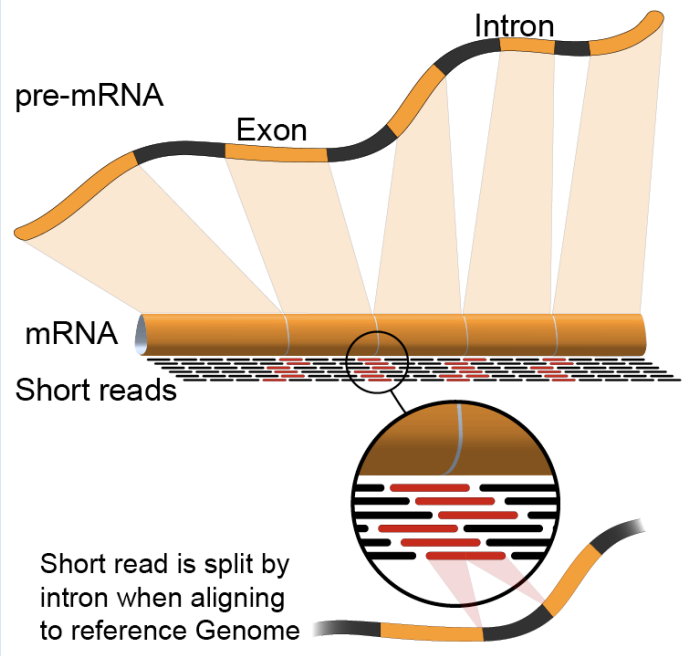
RNA-Seq Module 3
Abundance Estimation

My Hoang

Bfx workshop, December 4, 2023

Slides adapted from CSHL SeqTec course RNA-seq lecture series by Obi Griffith & Malachi Griffith





Washington University in St. Louis
School of Medicine

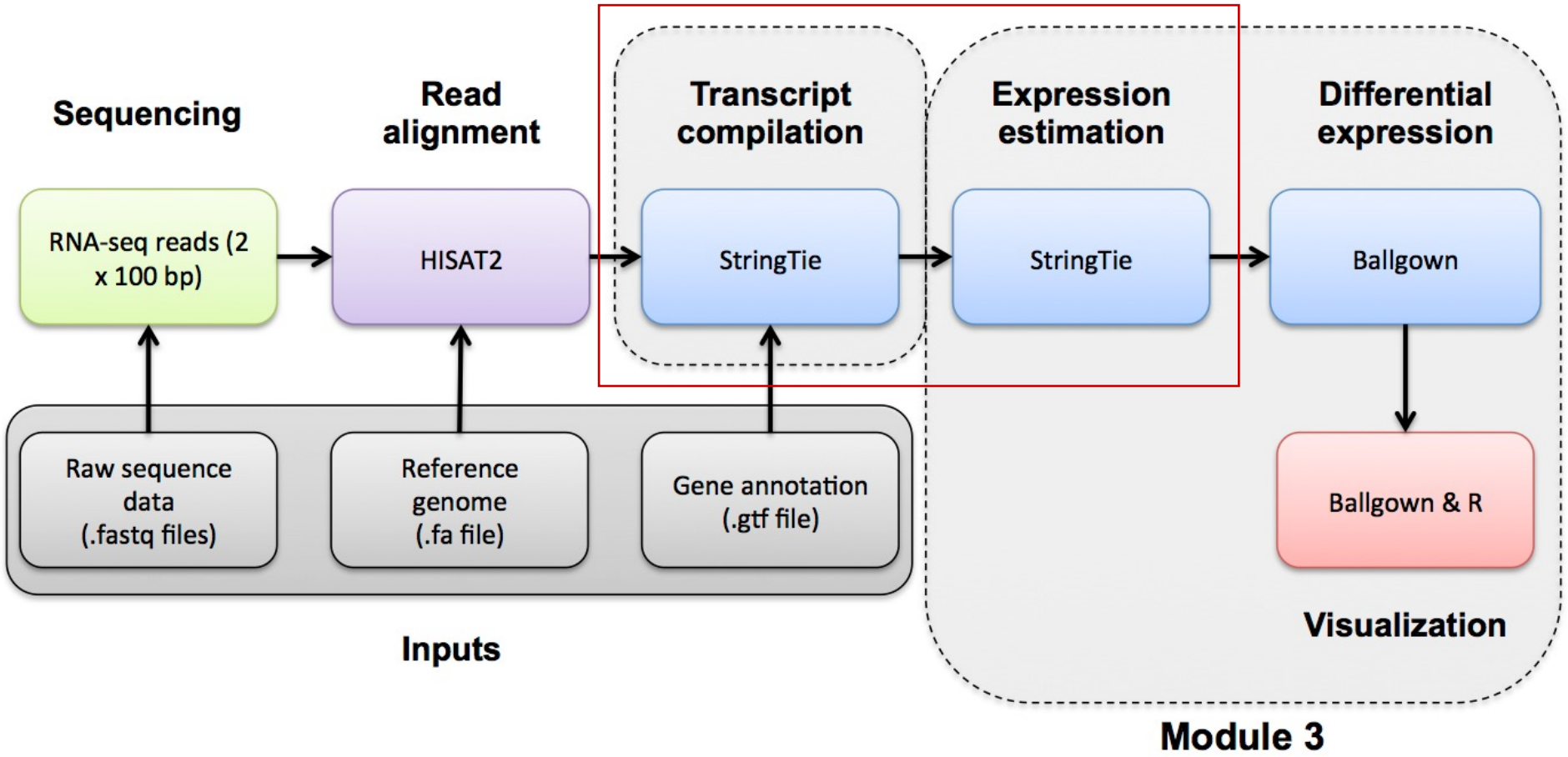
To-do

- Open docker desktop app in the background. Then in terminal, type:

$ docker pull griffithlab/rnabio:0.0.1

- Prepare alignment results (if you are stuck, download from http://genomedata.org/rnaseq-tutorial/results/cshl2022/rnaseq.tar.gz)

Overview

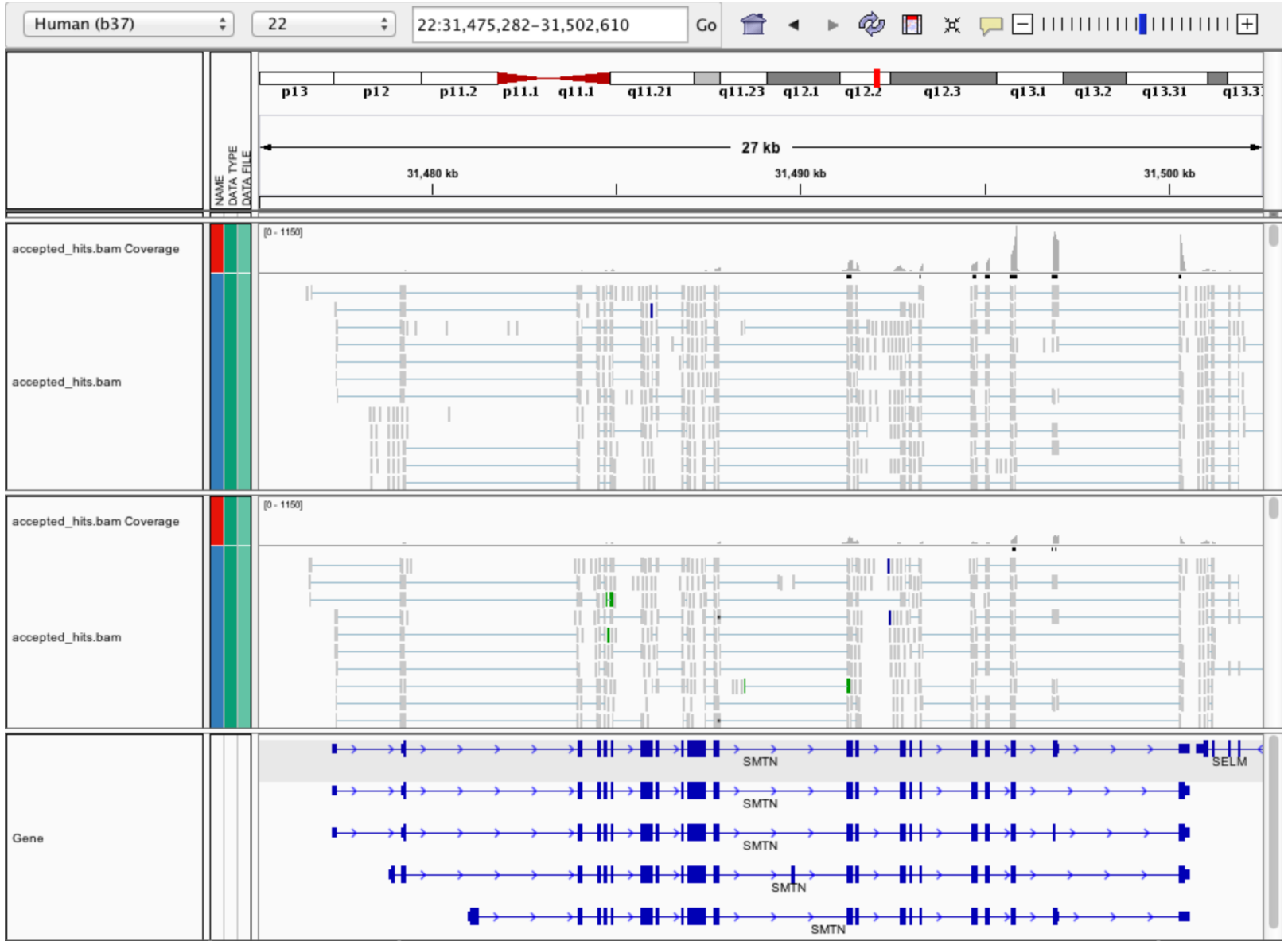


Last week: Alignment (HISAT2)
This week: Expression estimation (StringTie, htseq count)
… Next week: Differential expression (Ballgown, edgeR)

Learning Objectives of Module 3

- Review basic concepts and popular metrics of abundance estimation
- Review StringTie estimation approach and options
- Discuss raw read count approaches

Expression estimation for known genes and transcripts



3' bias

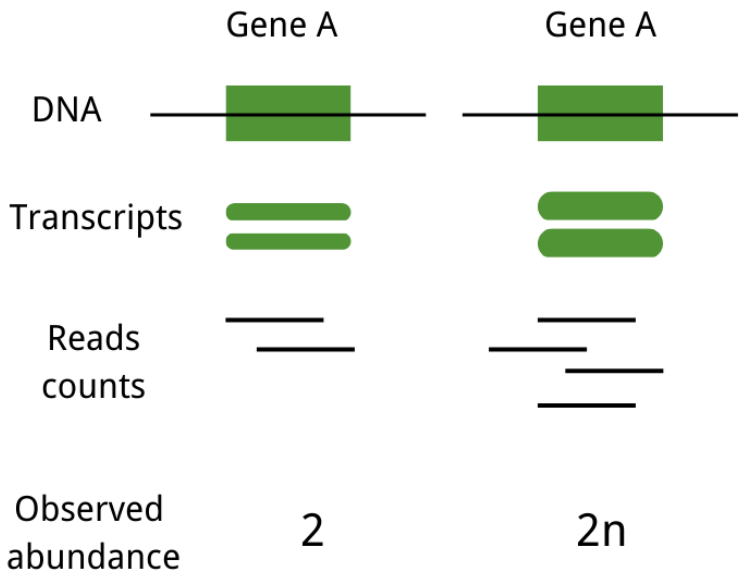
Down-regulated

Popular metrics for abundance estimation

- Raw counts
- Normalized counts:
  - RPKM, FPKM
  - TPM

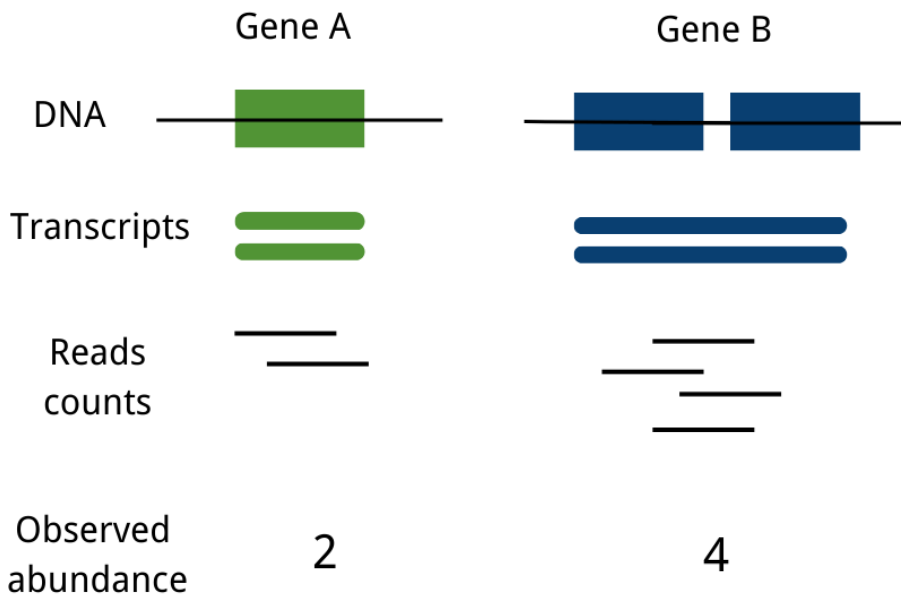
Normalized counts combat technical biases in sequenced data

- Sequencing depth



Sample with higher sequencing depth has more reads
→ Divided by mapped reads

- Gene length



Longer gene (gene B) has more reads
→ Divided by gene(transcript) length

Image adapted from novogene

What is FPKM (RPKM)?

- RPKM: **Reads** Per Kilobase of transcript per Million mapped reads.
- FPKM: **Fragments** Per Kilobase of transcript per Million mapped reads.
- Similar concept, RPKM is for single-end reads, FPKM is for paired-end reads

What is FPKM (RPKM)?

- RPKM: **Reads** Per Kilobase of transcript per Million mapped reads.
- FPKM: **Fragments** Per Kilobase of transcript per Million mapped reads.
- Similar concept, RPKM is for single-end reads, FPKM is for paired-end reads

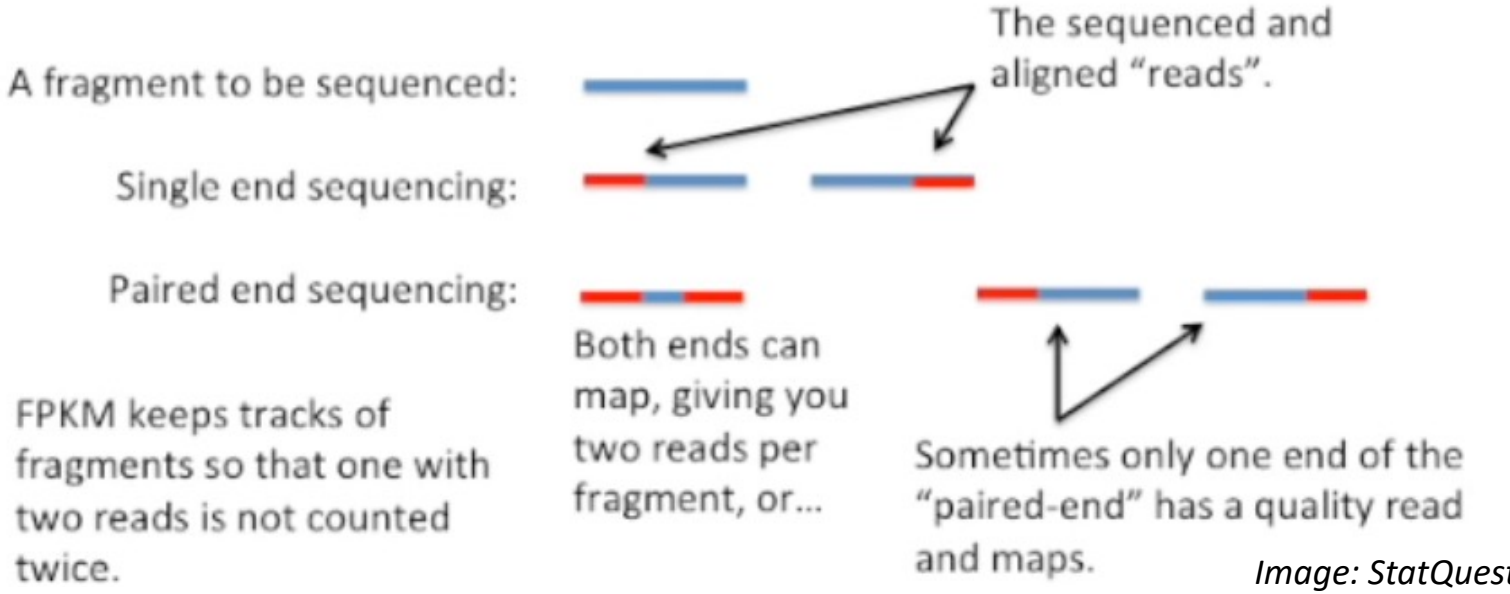
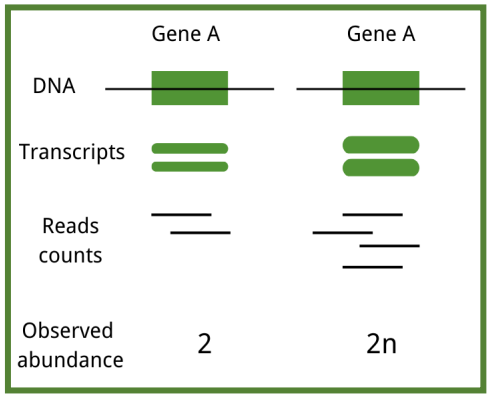
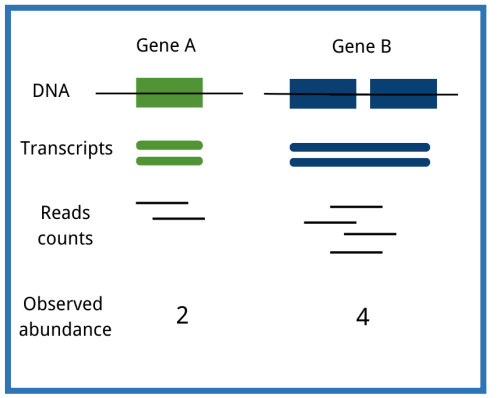


Image: StatQuest

What is FPKM?

- FPKM attempts to normalize for library depth and gene size
  - remember – RPKM is essentially the same!





- C = number of mappable fragments for a gene (transcript)
- N = total number of mappable fragments in the library
- L = number of base pairs in the gene (transcript)
  - FPKM = (C / (N / 1,000,000)) / (L/1000)

Per Million mapped reads **Per Kilobase of transcript**

= (10^9 x C) / (N x L)

- More reading:
  - http://www.biostars.org/p/11378/
  - http://www.biostars.org/p/68126/

How do FPKM and TPM differ?

- TPM: Transcript per Kilobase Million
- The difference is in the order of operations:

FPKM

1) Determine total fragment count, divide by 1,000,000 (per Million)

2) Divide each gene/transcript fragment count by #1 (Fragments Per Million)

3) Divide each FPM by length of each gene/transcript in kilobases (FPKM)

Normalize for sequencing depth, then normalize for gene length

TPM

1) Divide each gene/transcript fragment count by length of the transcript in kilobases (Fragments Per Kilobase)

2) Sum all FPK values for the sample and divide by 1,000,000 (per Million)

3) Divide #1 by #2 (TPM)

Normalize for gene length, then normalize for sequencing depth

- The sum of all TPMs in each sample is the same. Easier to compare across samples!

- http://www.rna-seqblog.com/rpkm-fpkm-and-tpm-clearly-explained/
- https://www.ncbi.nlm.nih.gov/pubmed/22872506

RPKM – step 1: normalize for read depth.

Gene Name	Rep1 Counts	Rep2 Counts	Rep3 Counts
A (2kb)	10	12	30
B (4kb)	20	25	60
C (1kb)	5	8	15
D (10kb)	0	0	1
Total reads:	35	45	106
Tens of reads:	3.5	4.5	10.6

For the purpose of this 4 gene example, we're scaling the total read counts by 10 instead of 1,000,000.

RPKM – step 2: normalize for gene length.

Gene Name	Rep1 RPM	Rep2 RPM	Rep3 RPM
A (2kb)	2.86	2.67	2.83
B (4kb)	5.71	5.56	5.66
C (1kb)	1.43	1.78	1.42
D (10kb)	0	0	0.09

Reads are scaled for depth (M) and gene length (K).

Gene Name	Rep1 RPKM	Rep2 RPKM	Rep3 RPKM
A (2kb)	1.43	1.33	1.42
B (4kb)	1.43	1.39	1.42
C (1kb)	1.43	1.78	1.42
D (10kb)	0	0	0.009

TPM – step 1: normalize for gene length

Original data:

Gene Name	Rep1 Counts	Rep2 Counts	Rep3 Counts
A (2kb)	10	12	30
B (4kb)	20	25	60
C (1kb)	5	8	15
D (10kb)	0	0	1

RPK – scaled by gene length:

Gene Name	Rep1 RPK	Rep2 RPK	Rep3 RPK
A (2kb)	5	6	15
B (4kb)	5	6.25	15
C (1kb)	5	8	15
D (10kb)	0	0	0.1

TPM – step 2: normalize for sequencing depth

Gene Name	Rep1 RPK	Rep2 RPK	Rep3 RPK
A (2kb)	5	6	15
B (4kb)	5	6.25	15
C (1kb)	5	8	15
D (10kb)	0	0	0.1
Total RPK:	15	20.25	45.1
Tens of RPK:	1.5	2.025	4.51

TPM – scaled by gene length and sequencing depth (M):

Gene Name	Rep1 TPM	Rep2 TPM	Rep3 TPM
A (2kb)	3.33	2.96	3.326
B (4kb)	3.33	3.09	3.326
C (1kb)	3.33	3.95	3.326
D (10kb)	0	0	0.02

Image: StatQuest

https://www.youtube.com/watch?v=TTUrtCY2k-w

RPKM vs TPM

RPKM

Gene Name	Rep1 RPKM	Rep2 RPKM	Rep3 RPKM
A (2kb)	1.43	1.33	1.42
B (4kb)	1.43	1.39	1.42
C (1kb)	1.43	1.78	1.42
D (10kb)	0	0	0.009
Total:	4.29	4.5	4.25

... the sums of each column are very different.

TPM

Gene Name	Rep1 TPM	Rep2 TPM	Rep3 TPM
A (2kb)	3.33	2.96	3.326
B (4kb)	3.33	3.09	3.326
C (1kb)	3.33	3.95	3.326
D (10kb)	0	0	0.02
Total:	10	10	10

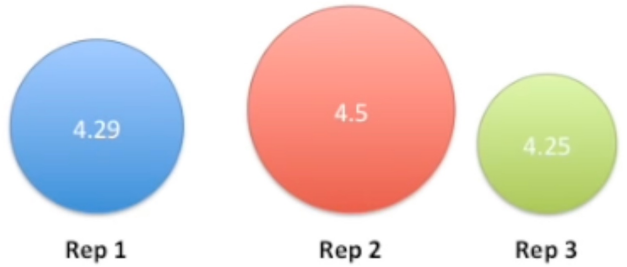
$$TPM = \frac{RPKM}{\sum(RPKM)} \times 10^6$$

RPKM

Gene Name	Rep1 RPKM	Rep2 RPKM	Rep3 RPKM
A (2kb)	1.43	1.33	1.42
B (4kb)	1.43	1.39	1.42
C (1kb)	1.43	1.78	1.42
D (10kb)	0	0	0.009
Total:	4.29	4.5	4.25

With RPKM, it is harder to compare the proportion of total reads because each replicate has different total (each pie has a different size)

A 1.43 size slice represents a different proportion of reads in in different pies.



Consider 3 pies, each the same size (10).

A 3.33 sized slice is the same in each pie, and is always larger than 3.32.

TPM makes it clear that in Rep1, more of its total reads mapped to gene A than in Rep3.

TPM

Gene Name	Rep1 TPM	Rep2 TPM	Rep3 TPM
A (2kb)	3.33	2.96	3.326
B (4kb)	3.33	3.09	3.326
C (1kb)	3.33	3.95	3.326
D (10kb)	0	0	0.02
Total:	10	10	10

Image: StatQuest

Summary

- Normalized counts account for sequencing depth and gene length biases
  - RPKM ~ single-end sequencing, FPKM ~ paired-end sequencing
  - The sum of all TPMs in each sample is the same. Easier to compare across samples!

Learning Objectives of Module 3

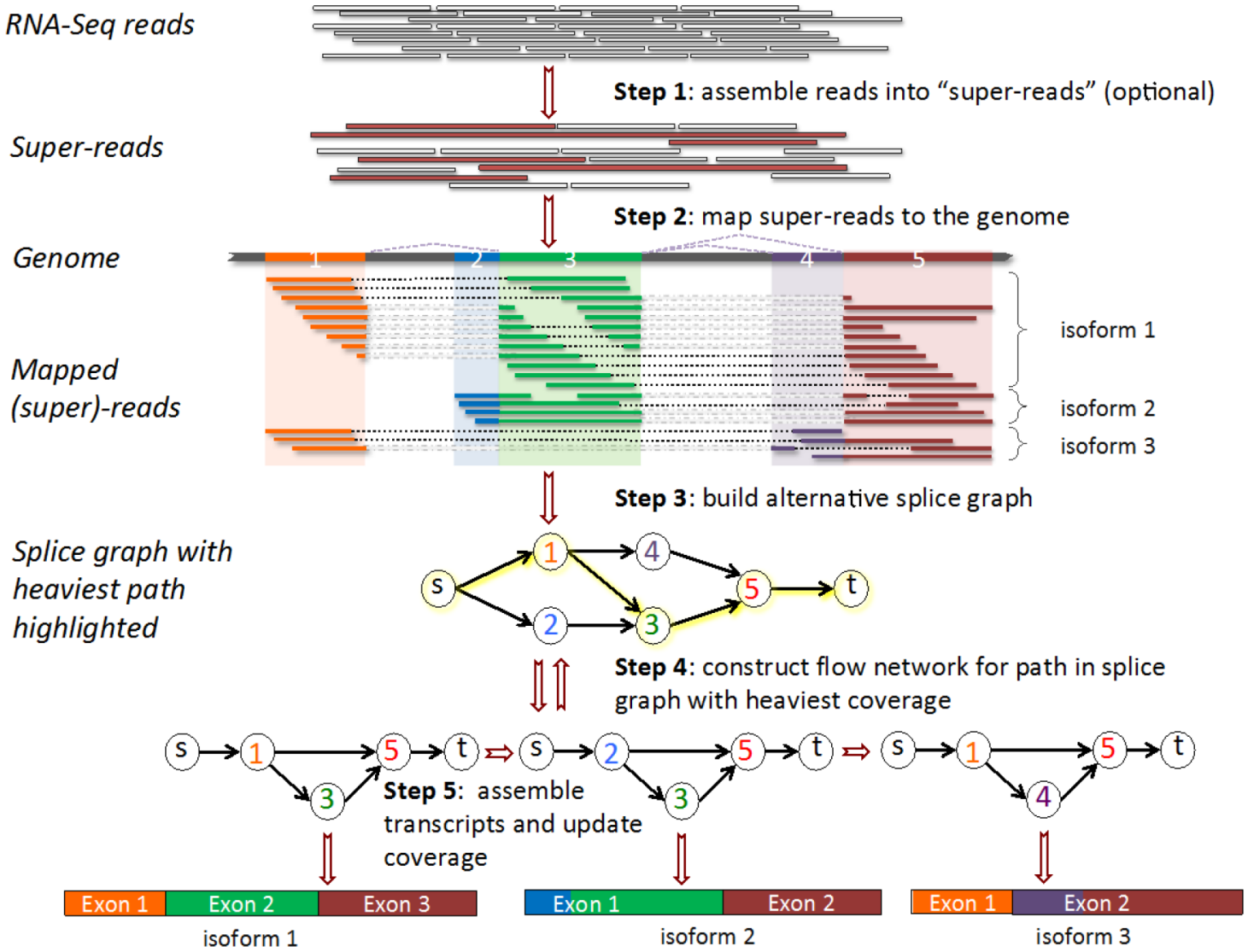
- Review basic concepts and popular metrics of abundance estimation
- Review StringTie estimation approach and options
- Discuss raw read count approaches

How does StringTie work?

- Align reads to the genome, optionally assemble super-reads and re-align
- Group reads into clusters

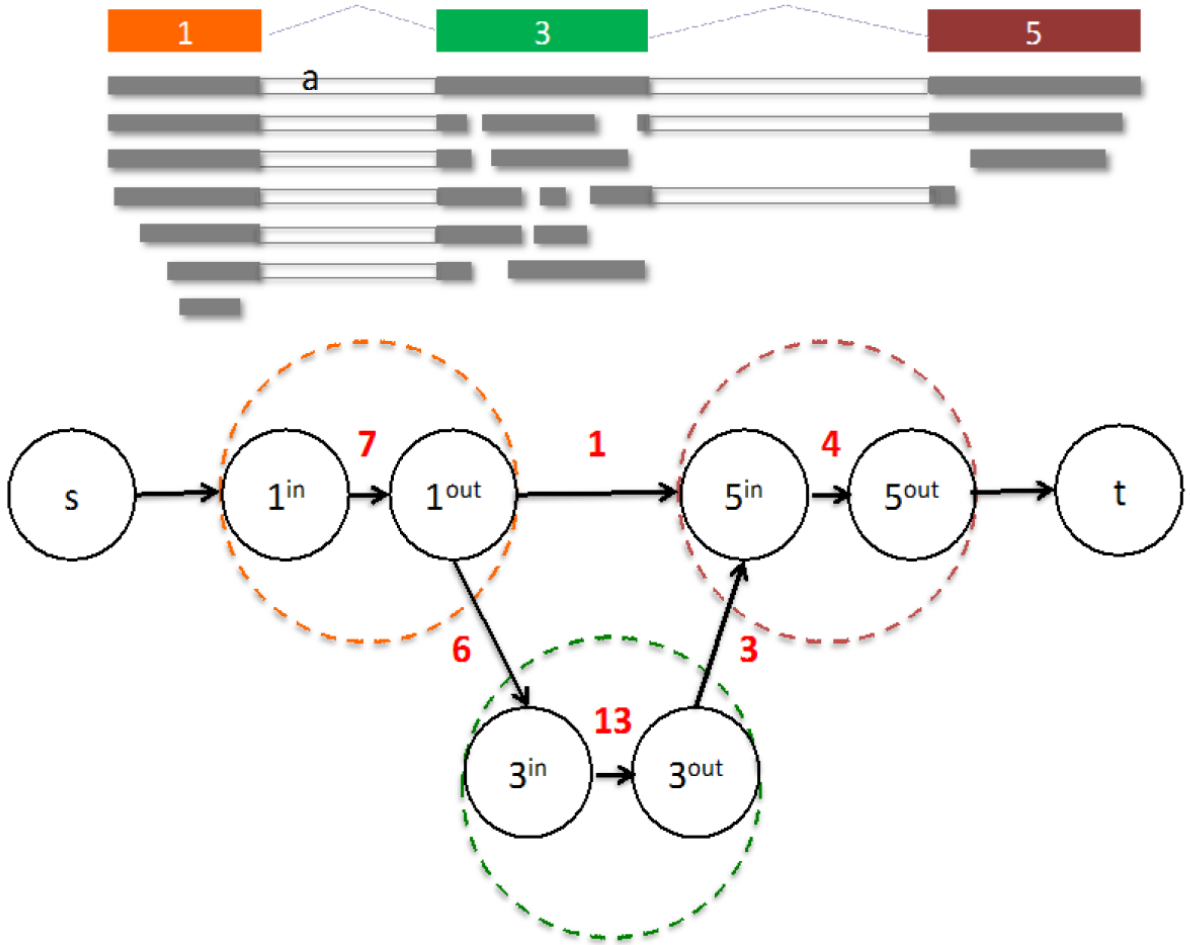
Infer isoforms:

- Build alternative splice graph (ASG)
- Iteratively extract the heaviest path from a splice graph
- construct a flow network
- compute maximum flow to estimate abundance
- update the splice graph by removing reads that were assigned by the flow algorithm
- This process repeats until all reads have been assigned.



Pertea et al. Nature Biotechnology, 2015

From flow network for each transcript, maximum flow is used to assemble transcript and estimate abundance



StringTie uses basic graph theory (splice graph), custom heuristics (heaviest path), more graph theory (flow network) and optimization theory (maximum flow). See StringTie paper for definitions and math.

StringTie Modes

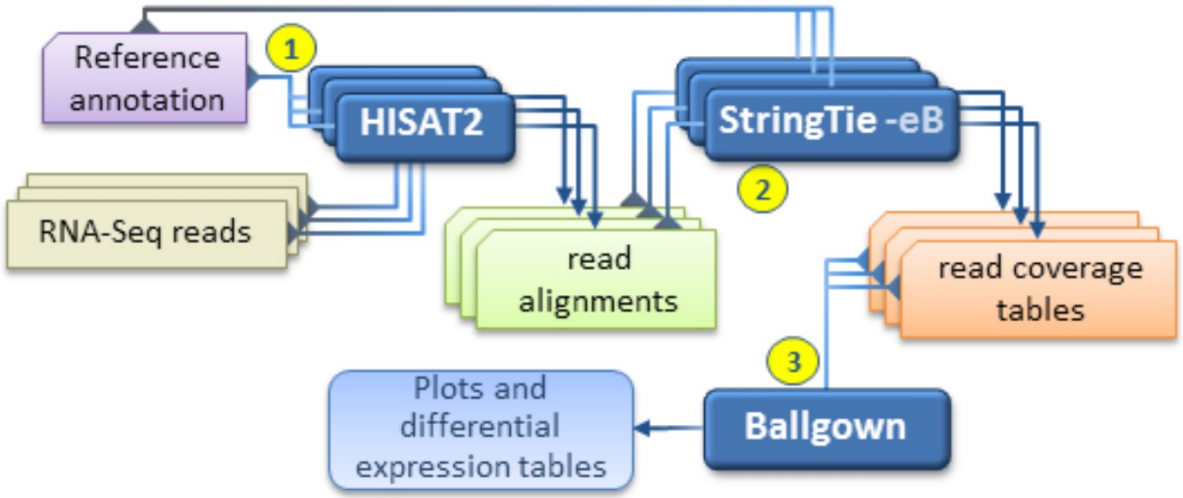
- Expression estimation mode (“Reference Only”) ← What we will use
  - “–G $GTF_File” AND “–e” option
  - no "novel" transcript assemblies (isoforms)
  - read alignments not overlapping reference transcripts ignored
  - Faster, especially when given limited set of reference transcripts
    - Avoids complicated steps of clustering and building alternative splice graph from scratch, skipping straight to abundance estimation
- “Reference guided mode”
  - “–G $GTF_File” WITHOUT “–e” option
  - Both known and novel transcript assemblies
- “De novo” mode
  - NEITHER “–G $GTF_File” NOR “–e” option
  - Novel transcript assemblies only

Pertea et al. Nature Protocols, 2016

StringTie -merge

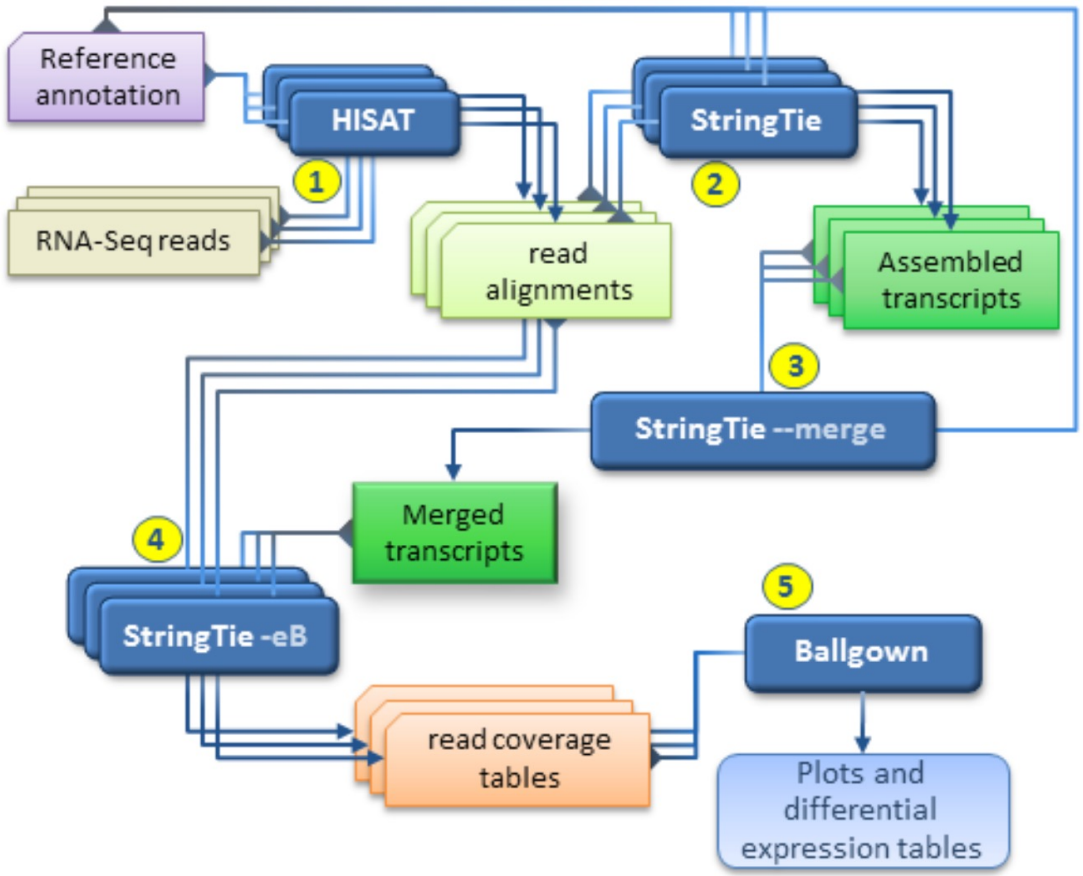
- Merge together all gene structures from all samples
  - Some samples may only partially represent a gene structure
- Incorporates known transcripts with assembled, potentially novel transcripts
- For de novo or reference guided mode, we will rerun StringTie with the merged transcript assembly.

Pertea et al. Nature Protocols, 2016



This is the workflow we use in the exercise: (bypass StringTie --merge,) use StringTie –G and -e

Expression estimation mode (“Reference Only”)



But in case you want to run Reference-guide or ‘Denovo’ mode, will need to run StringTie, then StringTie --merge, then StringTie –e .

https://ccb.jhu.edu/software/stringtie/index.shtml?t=manual

gffcompare

- gffcompare will compare a merged transcript GTF with known annotation, also in GTF/GFF3 format
- https://ccb.jhu.edu/software/stringtie/gff.shtml#gffcompare

Priority	Code	Description
1	=	Complete match of intron chain
2	c	Contained
3	j	Potentially novel isoform (fragment): at least one splice junction is shared with a reference transcript
4	e	Single exon transfrag overlapping a reference exon and at least 10 bp of a reference intron, indicating a possible pre-mRNA fragment.
5	i	A transfrag falling entirely within a reference intron
6	o	Generic exonic overlap with a reference transcript
7	p	Possible polymerase run-on fragment (within 2Kbases of a reference transcript)
8	r	Repeat. Currently determined by looking at the soft-masked reference sequence and applied to transcripts where at least 50% of the bases are lower case
9	u	Unknown, intergenic transcript
10	x	Exonic overlap with reference on the opposite strand
11	s	An intron of the transfrag overlaps a reference intron on the opposite strand (likely due to read mapping errors)
12	.	(.tracking file only, indicates multiple classifications)

Alternatives to FPKM

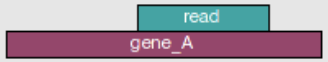
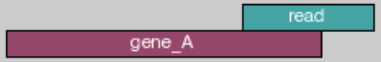
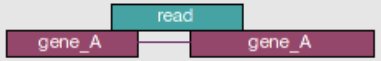
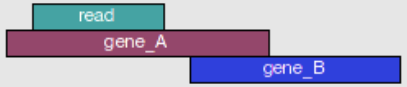
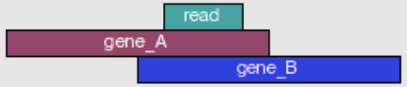
- Raw read counts for differential expression analysis
  - Assign reads/fragments to defined genes/transcripts, get “raw counts”
    - Transcript structures could still be defined by something like Stringtie

- HTSeq (htseq-count)
  - https://htseq.readthedocs.io/

```
htseq-count --mode intersection-strict --stranded no --minaqual 1 --type exon --idattr transcript_id
accepted_hits.sam chr22.gff > transcript_read_counts_table.tsv
```

- Caveats of ‘transcript’ analysis by htseq-count:
  - Designed for genes - ambiguous reads from overlapping transcripts may not be handled!
  - http://seqanswers.com/forums/showthread.php?t=18068

HTSeq-count basically counts reads supporting a feature (exon, gene) by assessing overlapping coordinates

	union	intersection _strict	intersection _nonempty
	gene_A	gene_A	gene_A
	gene_A	no_feature	gene_A
	gene_A	no_feature	gene_A
	gene_A	gene_A	gene_A
	gene_A	gene_A	gene_A
	ambiguous	gene_A	gene_A
	ambiguous	ambiguous	ambiguous

Note, if gene_A and gene_B on opposite strands, sequence data is stranded, and correct HTSeq parameter set then this read may not be ambiguous

Whether a read is counted depends on the nature of overlap and “mode” selected

Summary

- Normalized counts account for sequencing depth and gene length biases
  - RPKM ~ single-end sequencing, FPKM ~ paired-end sequencing
  - The sum of all TPMs in each sample is the same. Easier to compare across samples!
- Abundance estimation tool that calculates normalized count (FPKM, TPM): StringTie
- Abundance estimation tool that calculates raw count: HTseq

We are on a Coffee Break & Networking Session

Set up

- Open docker desktop app in the background. Then in terminal, type:
- $ docker pull griffithlab/rnabio:0.0.1
- $ cd bfx-workshop/rnabio-workspace
- $ docker run -v $PWD/:/workspace:rw -it griffithlab/rnabio:0.0.1 /bin/bash

(or in case it throws error

- $ docker run --platform linux/amd64 -v $PWD/:/workspace:rw -it griffithlab/rnabio:0.0.1 /bin/bash

)

- $ cd workspace
- $ su ubuntu
- $ source ~/.bashrc

Strandedness

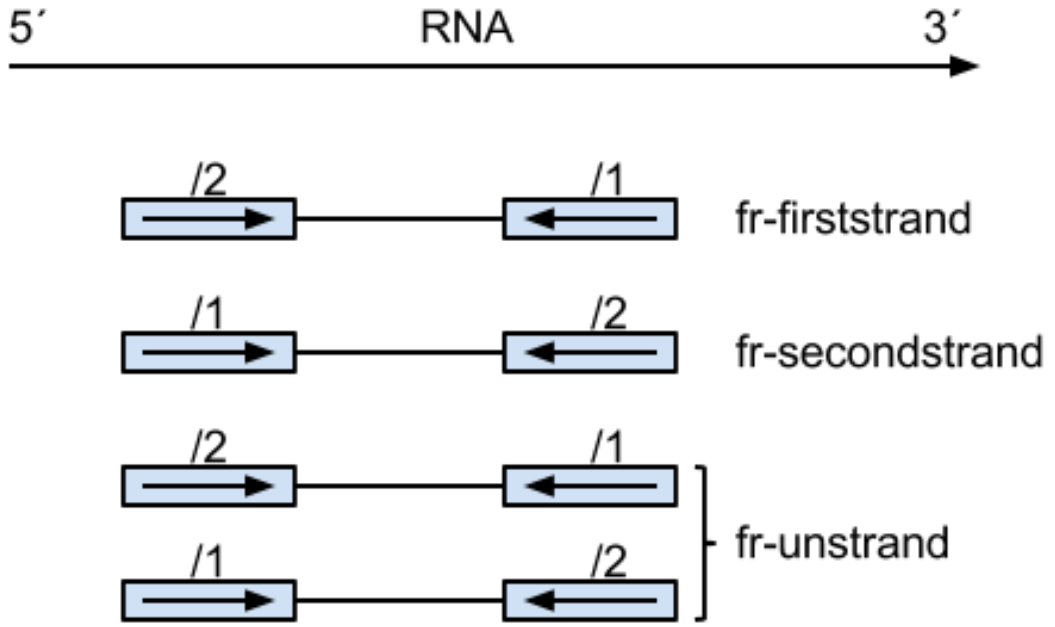


Figure:
+ The reads on the left are from the same strand as the transcript, and their pairs on the right are from the opposing strand.

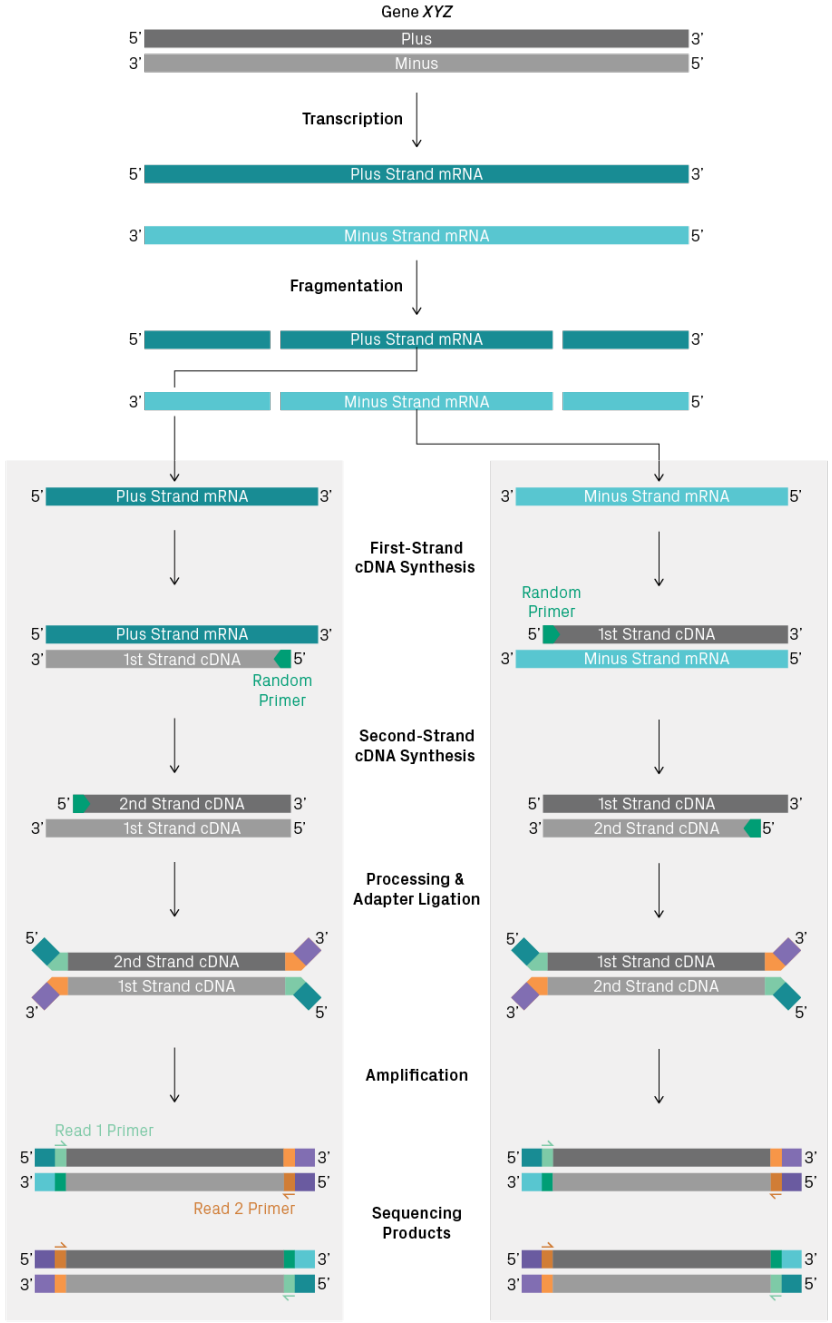
RF/fr-firststrand stranded (dUTP)	FR/fr-secondstrand stranded (Ligation)	Unstranded
The second read (read 2) is from the original RNA strand/template, first read (read 1) is from the opposite strand	The first read (read 1) is from the original RNA strand/template, second read (read 2) is from the opposite strand.	Information regarding the strand is not conserved (it is lost during the amplification of the mRNA fragments).

Why is this so important?

- If you use wrong directionality parameter in the read counting step with HTSeq, the reads are considered to be from the wrong strand. This means that in the case where there is no gene on that other strand, you won't get any counts, and if there is a gene in the same location on the other strand, your reads are counted for the wrong gene.
- If you use wrong directionality parameter in the reference alignment step, the XS tag in the resulting BAM file will contain wrong strand information. The XS tag is used by transcript assembly programs like Cufflinks and Stringtie, and also Cuffdiff uses it.

https://chipster.csc.fi/manual/library-type-summary.html#:~:text=second%2Dstrand%20%3D%20directional%2C%20where,is%20from%20the%20opposite%20strand.

Non-Stranded Library Prep



https://www.azenta.com/blog/stranded-versus-non-stranded-rna-seq

Stranded Library Prep

Library Preparation Selection Guide

Choosing the right library preparation method depends on several factors, including your experimental objective, budget, and availability of a reference transcriptome for your organism.

The most important consideration is the objective of your experiment. Stranded RNA-Seq is strongly recommended if you're trying to accomplish one or more of the following, as it's important to capture information about transcript directionality:

- Identify antisense transcripts
- Annotate a genome
- Discover novel transcripts

Non-stranded RNA-Seq, on the other hand, is often sufficient for measuring gene expression in organisms with well-annotated genomes. With a reference transcriptome, you can infer orientation for most of the sequencing reads. Since there are fewer steps than stranded library prep, the benefits of this approach are lower cost, simpler execution, and greater recovery of material.

Also, when comparing the results of a new experiment to older ones, many researchers prefer using the same RNA-Seq approach. It enables an apples-to-apples comparison of the data.

Key Takeaways

- Stranded RNA-Seq enables you to determine RNA orientation from each sequencing read; this information can't be directly obtained from non-stranded approaches
- By differentiating the first and second strands of cDNA, stranded library preparation preserves the directionality of the RNA molecule
- Certain applications require a stranded approach; however, non-stranded RNA-Seq is suitable for many NGS projects

https://www.azenta.com/blog/stranded-versus-non-stranded-rna-seq

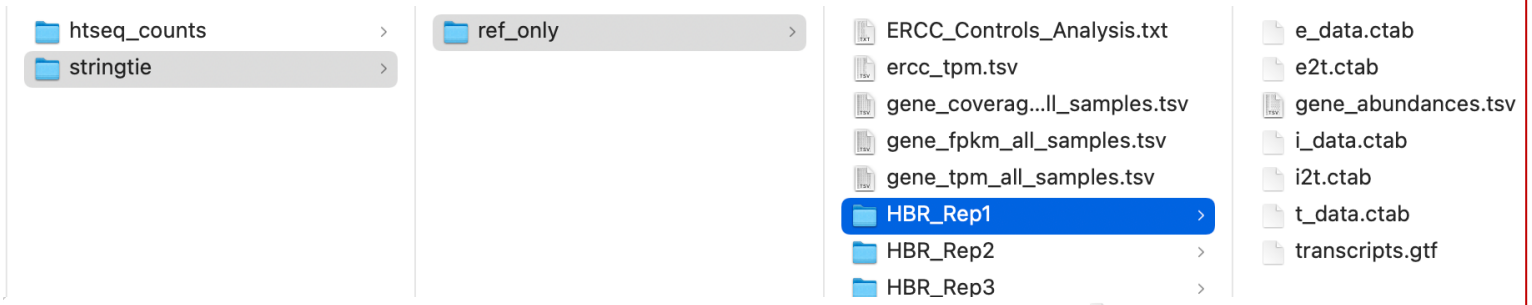
Stringtie outputs

- Stringtie gives 3 metrics for expression levels: coverage, FPKM, TPM ; for 2 types : transcript and gene.
- Focus on the ‘transcript.gtf’ and ‘gene_abundance.tsv’

+ 'e_data.ctab’ : exon coordinate and corresponding coverage
+ ‘i_data.ctab’: intron and corresponding coverage
+ ‘t_data.ctab’: transcript and corresponding coverage
+ e2t.ctab: table with two columns, e_id and t_id, denoting which exons belong to which transcripts
+ i2t.ctab: table with two columns, i_id and t_id, denoting which introns belong to which transcripts

+ transcripts.gtf : transcripts, the exons made up each transcript, coordinates, gene id/ transcript id, cov / fpkm / tpm

+ gene_abundance.tsv: only gene name, coordinates, and expr levels



transcripts.gtf

```
# stringtie --rf -p 8 -G /home/ubuntu/workspace/rnaseq/refs/chr22_with_ERCC92.gtf -e -B -o HBR_Rep1/transcripts.gtf -A HBR_Rep1/gene_abundances.tsv /home/ubuntu/workspace/rnaseq/alignments/hisat2/HBR_Rep1.bam
# StringTie version 2.1.6
22      havana  transcript      16869478 16871126 .       +       .       gene_id "ENSG00000237689"; transcript_id "ENST00000442403"; ref_gene_name "AC007064.24"; cov "0.0"; FPKM "0.000000"; TPM "0.000000";
22      havana  exon    16869478 16869626 .       +       .       gene_id "ENSG00000237689"; transcript_id "ENST00000442403"; exon_number "1"; ref_gene_name "AC007064.24"; cov "0.0";
22      havana  exon    16870776 16871126 .       +       .       gene_id "ENSG00000237689"; transcript_id "ENST00000442403"; exon_number "2"; ref_gene_name "AC007064.24"; cov "0.0";
22      havana  transcript      15790709 15791814 .       +       .       gene_id "ENSG00000206195"; transcript_id "ENST00000458623"; ref_gene_name "DUXAP8"; cov "0.0"; FPKM "0.000000"; TPM "0.000000";
22      havana  exon    15790709 15790798 .       +       .       gene_id "ENSG00000206195"; transcript_id "ENST00000458623"; exon_number "1"; ref_gene_name "DUXAP8"; cov "0.0";
22      havana  exon    15791017 15791152 .       +       .       gene_id "ENSG00000206195"; transcript_id "ENST00000458623"; exon_number "2"; ref_gene_name "DUXAP8"; cov "0.0";
22      havana  exon    15791628 15791814 .       +       .       gene_id "ENSG00000206195"; transcript_id "ENST00000458623"; exon_number "3"; ref_gene_name "DUXAP8"; cov "0.0";
```

e_data.ctab

16869478

```
264     22      +       16869478 16869626 0       0       0.00    0.0000  0.0000  0.0000  0.0000
265     22      +       16870776 16871126 2       2       2.00    0.5698  0.5992  0.5698  0.5992
```

t_data.ctab

```
t_id    chr     strand  start   end     t_name  num_exons       length  gene_id gene_name       cov     FPKM
1       22      -       10736171 10736283 ENST00000615943 1       113     ENSG00000277248 U2      0.000000 0.000000
2       22      -       10939388 10961338 ENST00000635667 9       749     ENSG00000283047 FRG1FP  0.000000 0.000000
```

gene_abundances.tsv

```
Gene ID Gene Name       Reference       Strand  Start   End     Coverage FPKM     TPM
ENSG00000237689 AC007064.24     22      +       16869478 16871126 0.000000 0.000000 0.000000
ENSG00000206195 DUXAP8  22      +       15784959 15829984 0.282236 52.024742        77.325455
```