

# RNA-Seq Alignment and Quantification

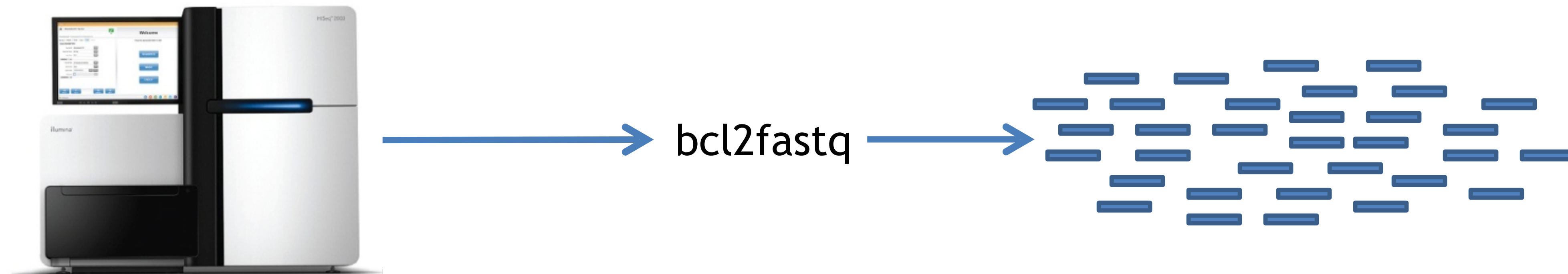
---

Turning sequence data into analyzable data

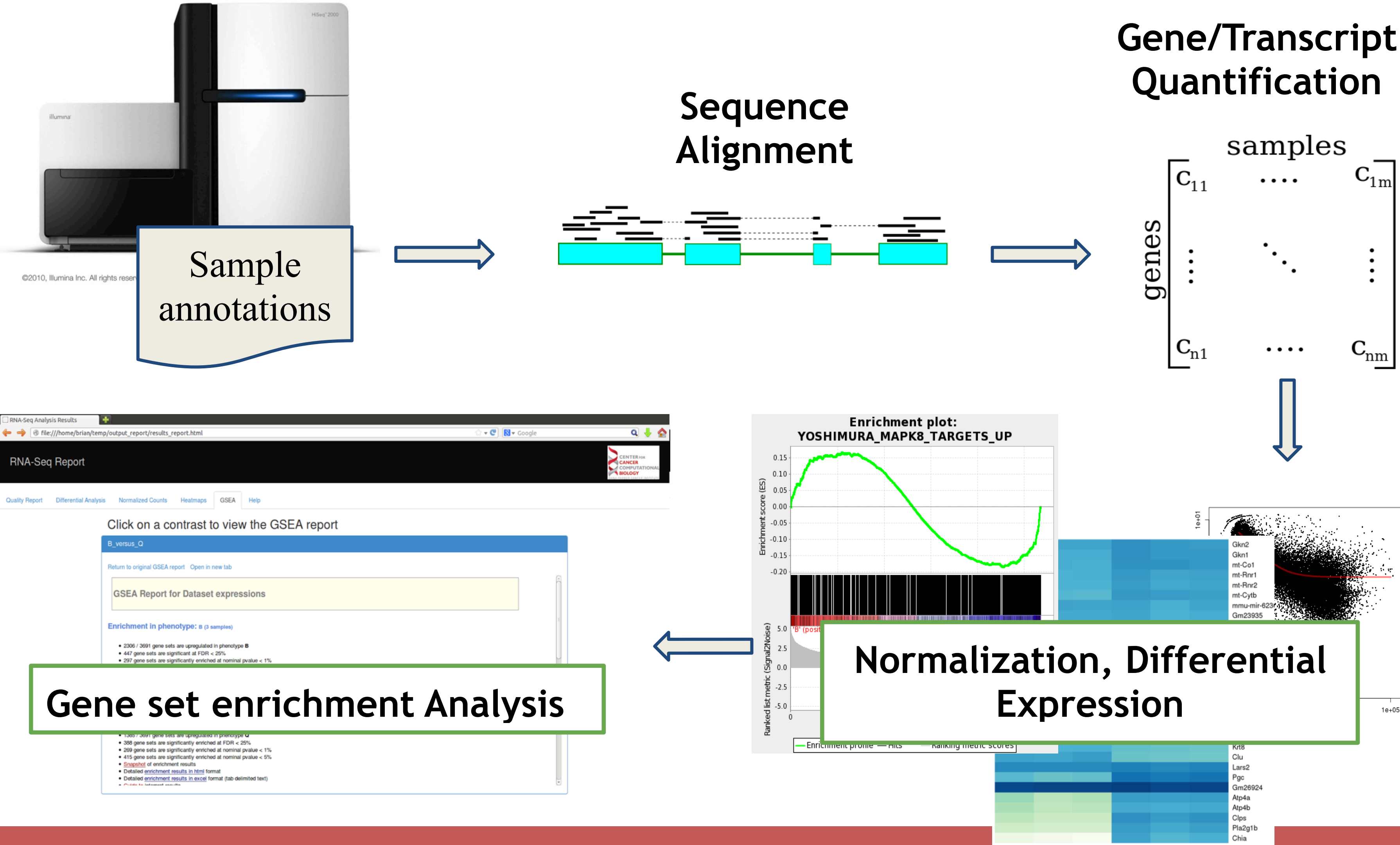
# Reads are ready. Now What?

## Big Fastq files (2-30Gb)

- Reads represent real biology.
- More reads corresponding to a transcript indicate higher abundance of that transcript.
- Reads may represent novel transcripts or novel arrangements of exons that are not present in any known reference genome.
- New exon-exon junctions, RNA-editing, and nucleotide variations (SNPs) may all be present in the read data.
- How do we translate these raw reads into biological knowledge: start with sequence alignment.



# RNA-Seq data analysis workflow for differential gene expression



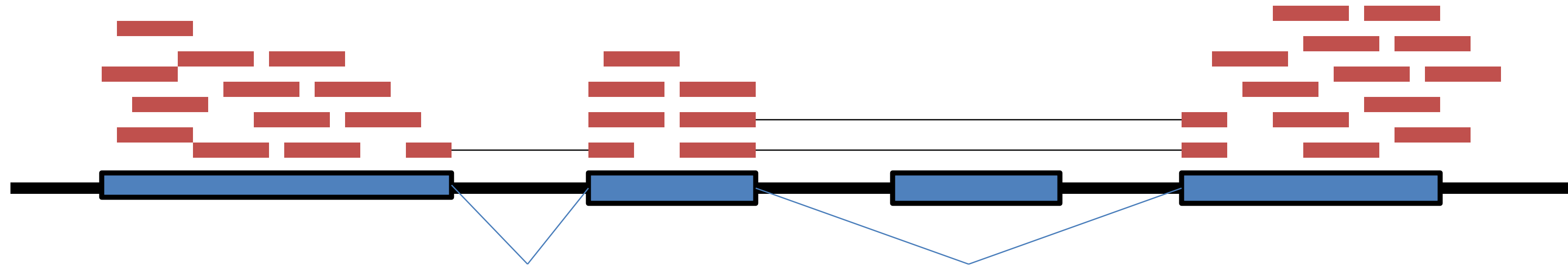
# Gene and Transcript Quantification

---

Counting reads and quantifying gene expression across different samples for comparison

# Map to the genome, with knowledge of transcript annotations

---



To effectively map to exon junctions, you need a mapping algorithm that can divide the sequencing reads and map portions independently.

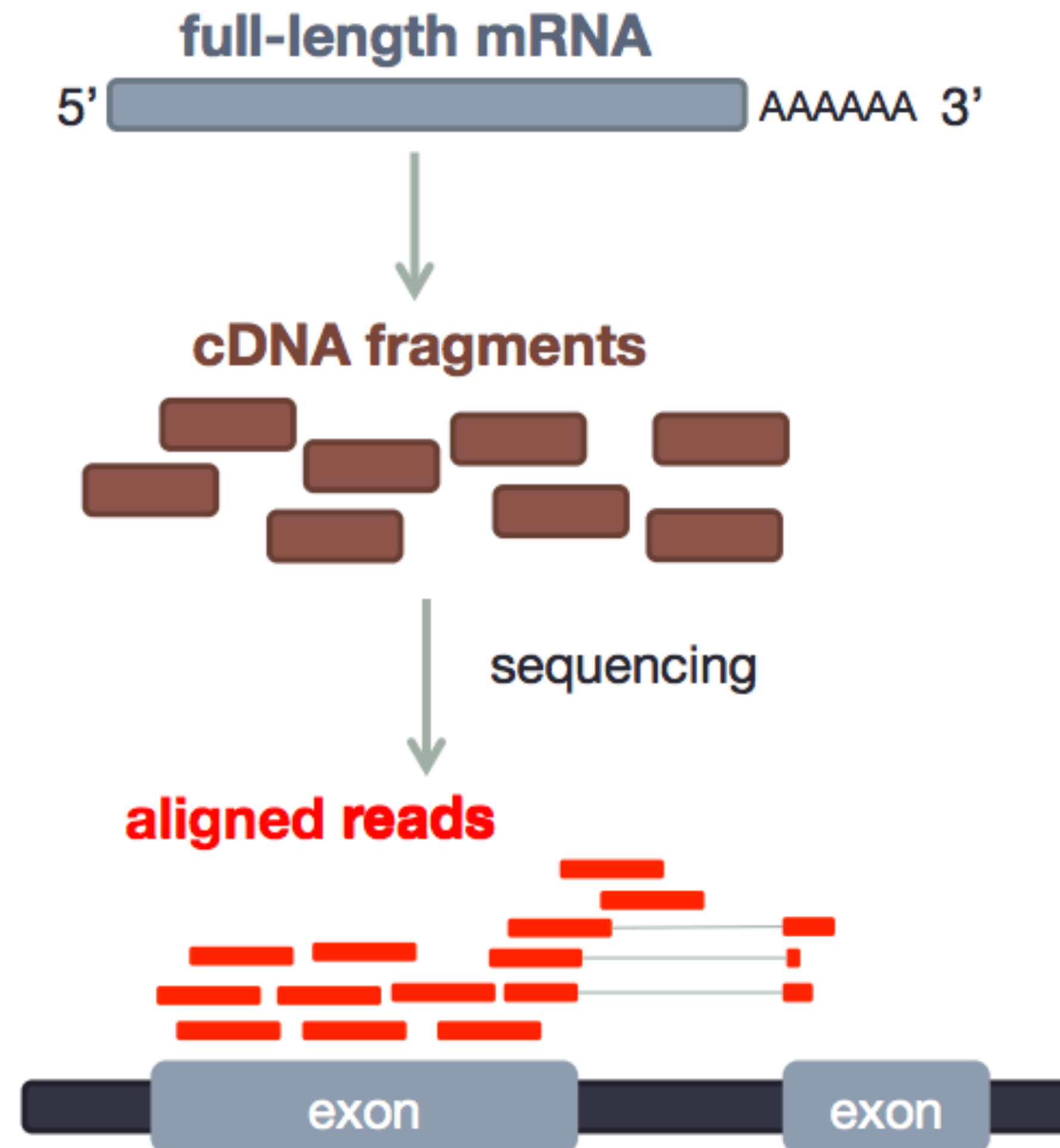
Identifying alternative transcript isoforms involves complex algorithms.

# Different philosophies of transcript quantification

---

- **Alignment-based:** Sequence alignment followed by counting of reads overlapping with a given annotated gene
- **Pseudo-Alignment:** Sometimes called 'Alignment Free', Quantify the number of reads that are consistent with a given transcript ( the exact location within the transcript is ignored)

# Aligning Short Read to Transcriptome

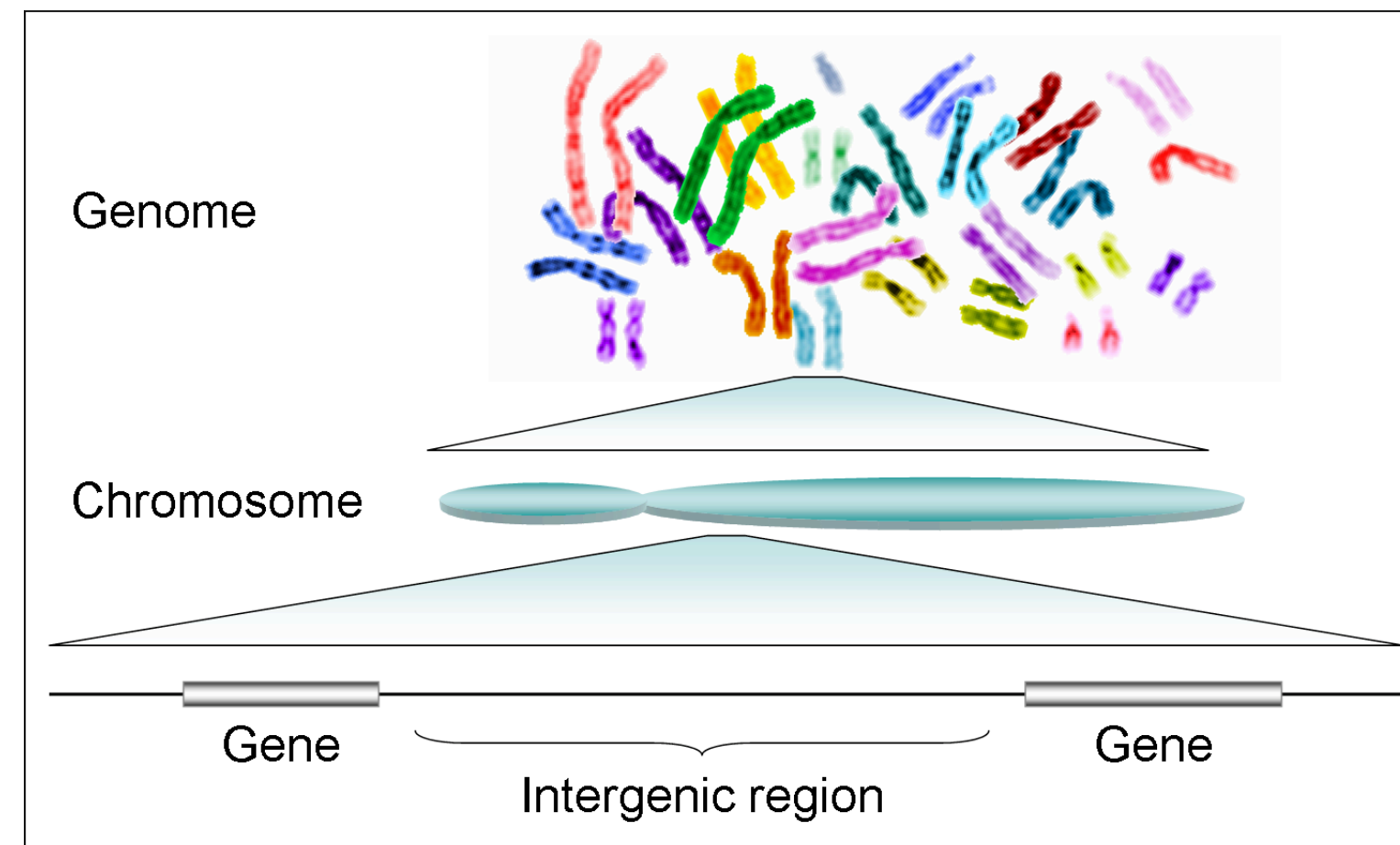
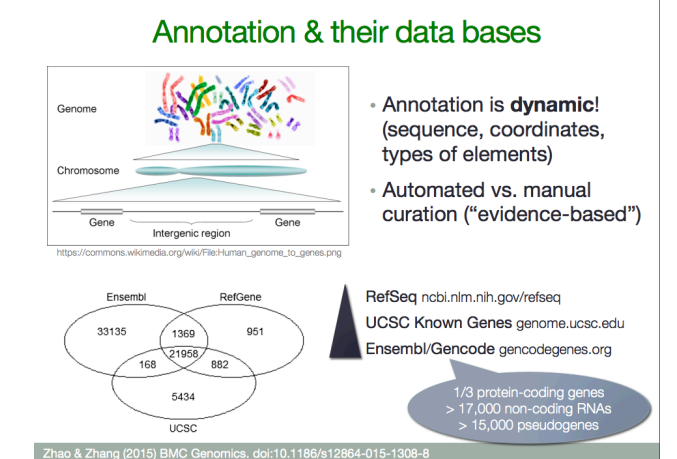


Short read alignment tools usually need:

1. Raw sequence data file in the form of fastq.gz
2. Reference genome sequence
3. Gene model annotation



# Reference Genome Annotation



- Human genome annotation is evolving! (Sequence, gene coordinates, types of elements)
- Automated vs manual curation (“evidence-based”)
- And from different databases:
  - **RefSeq** (<https://www.ncbi.nlm.nih.gov/refseq/>): Most stringent annotation criteria, less genes, but more sequence versions
  - **Ensembl/Gencode** (<http://ensembl.org>/<https://gencodegenes.org>): Almost the same



# Storing Annotation Information

Storing annotation information

• representing genome coordinates + description/name

• various formats (all are plain text files): GFF2, GFF3, GTF, BED, SAF...

see the course notes for details

GFF2

GTF ("GFF2.5")

GFF3

GTF

1. reference coordinate

2. source

3. annotation type

4. start position

5. end position

6. score

7. strand

8. frame/phase

9. attributes: <TYPE VALUE>; <TYPE VALUE>; <TYPE VALUE>

# GTF-version 2

chr12 . exon 5506900 5506996 . .

chr12 . exon 5506996 5507000 . .

chr12 . exon 5507000 5507004 . .

# GTF-version 3

chr12 . exon 1000 1000 . . ID=exon00001

chr12 . exon 1000 1000 . . ID=exon00002

chr12 . exon 3000 3000 . . ID=exon00003

chr12 . exon 8000 8000 . . ID=exon00004

chr12 . exon 7000 9000 . . ID=exon00005

example for the 9th field of a GTF file

gene\_id "En:062.C22.6"; transcript\_id "En:062.C22.6.mRNA"; exon\_number 1

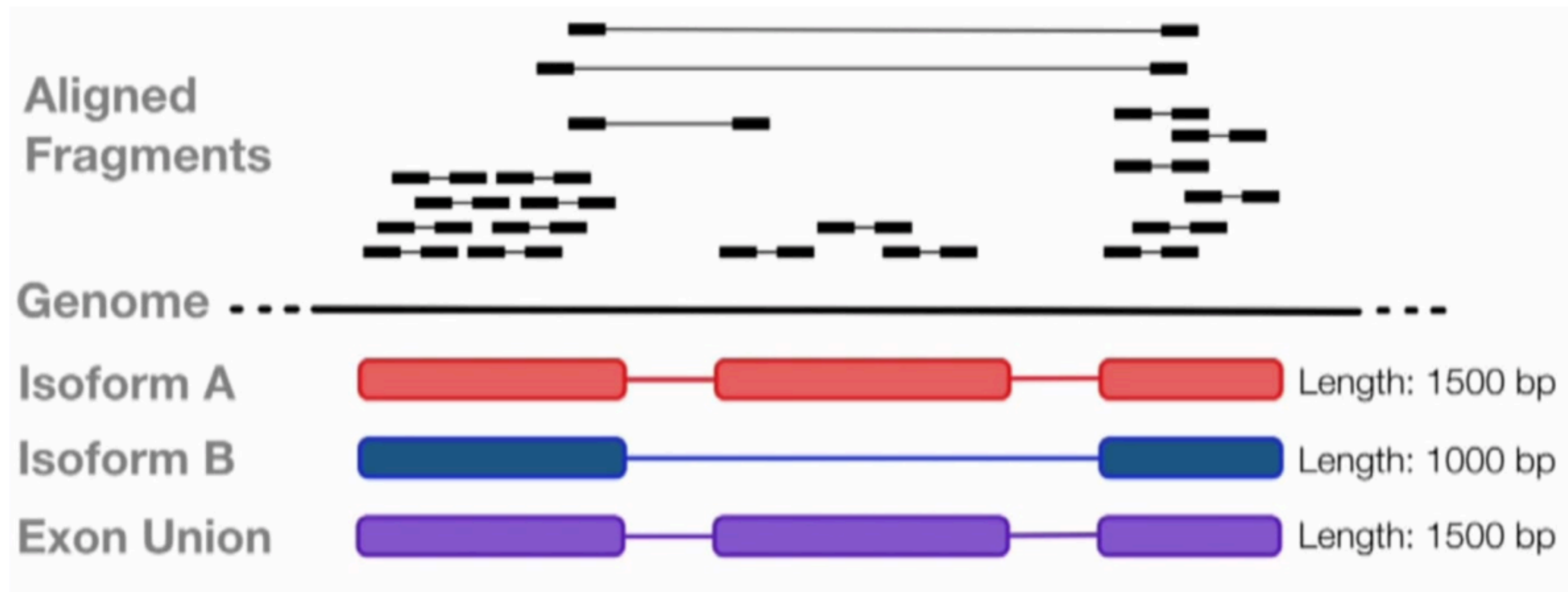
- Representing genome coordinates and description/name
- Gene and transcript annotations are stored in various formats: GFF2, GFF3, GTF, BED, ...
- Most commonly used format is GTF (“GFF2.5”)

## GTF Attributes

1. Reference coordinate
2. Source
3. Annotation type
4. Start position
5. End position
6. Score
7. Strand
8. frame/phase
9. Attributes: <TYPE VALUE>; <Type VALUE>; <TYPE VALUE>...

1	ensembl_havana	CDS	223985917	223986379	.	-	0	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";
1	ensembl_havana	exon	223983517	223984292	.	-	.	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";
1	ensembl_havana	CDS	223983517	223984292	.	-	2	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";
1	ensembl_havana	exon	223981005	223981142	.	-	.	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";
1	ensembl_havana	CDS	223981005	223981142	.	-	0	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";
1	ensembl_havana	exon	223980091	223980224	.	-	.	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";
1	ensembl_havana	CDS	223980091	223980224	.	-	0	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";
1	ensembl_havana	exon	223976710	223976876	.	-	.	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";
1	ensembl_havana	CDS	223976710	223976876	.	-	1	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";
1	ensembl_havana	exon	223971817	223972016	.	-	.	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";
1	ensembl_havana	CDS	223971817	223972016	.	-	2	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";
1	ensembl_havana	exon	223967601	223968596	.	-	.	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";
1	ensembl_havana	CDS	223968558	223968596	.	-	0	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";
1	ensembl_havana	stop_codon	223968555	223968557	.	-	0	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";
1	ensembl_havana	five_prime_utr	224033383	224033674	.	-	.	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";
1	ensembl_havana	three_prime_utr	223967601	223968554	.	-	.	gene_id "ENSG00000143514"; gene_version "12"; transcript_id "ENST00000343537";

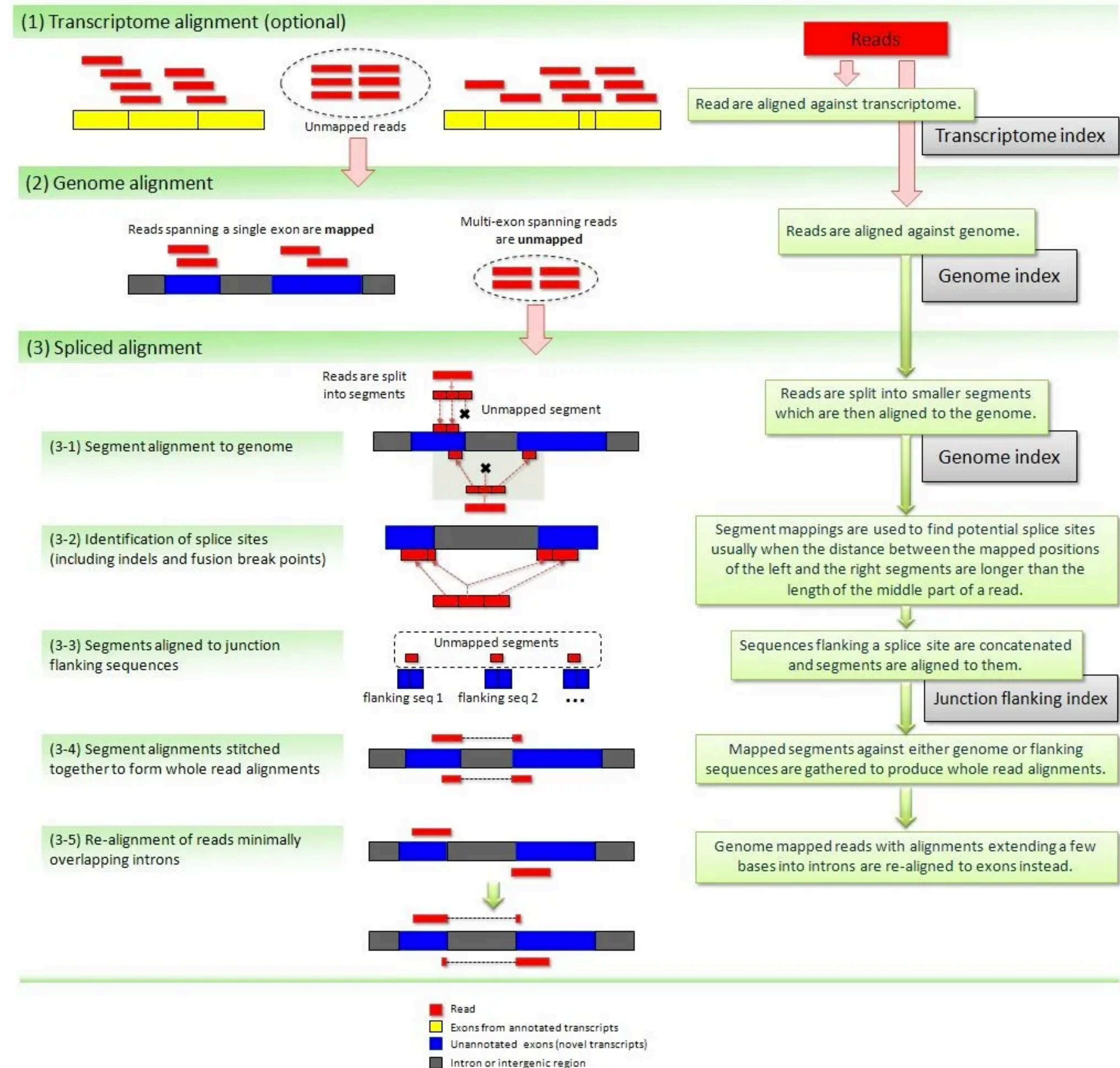
# Aligning to Transcript model





# Example 1: TopHat2

- Create genome index from reference genome index
- Map only reads spanning within a single exon (mapped reads)
- Rest of the reads (unmapped reads) are split into segments
- Paired segments are mapped individually to potential splice sites
- Mapped split reads are stitched back together





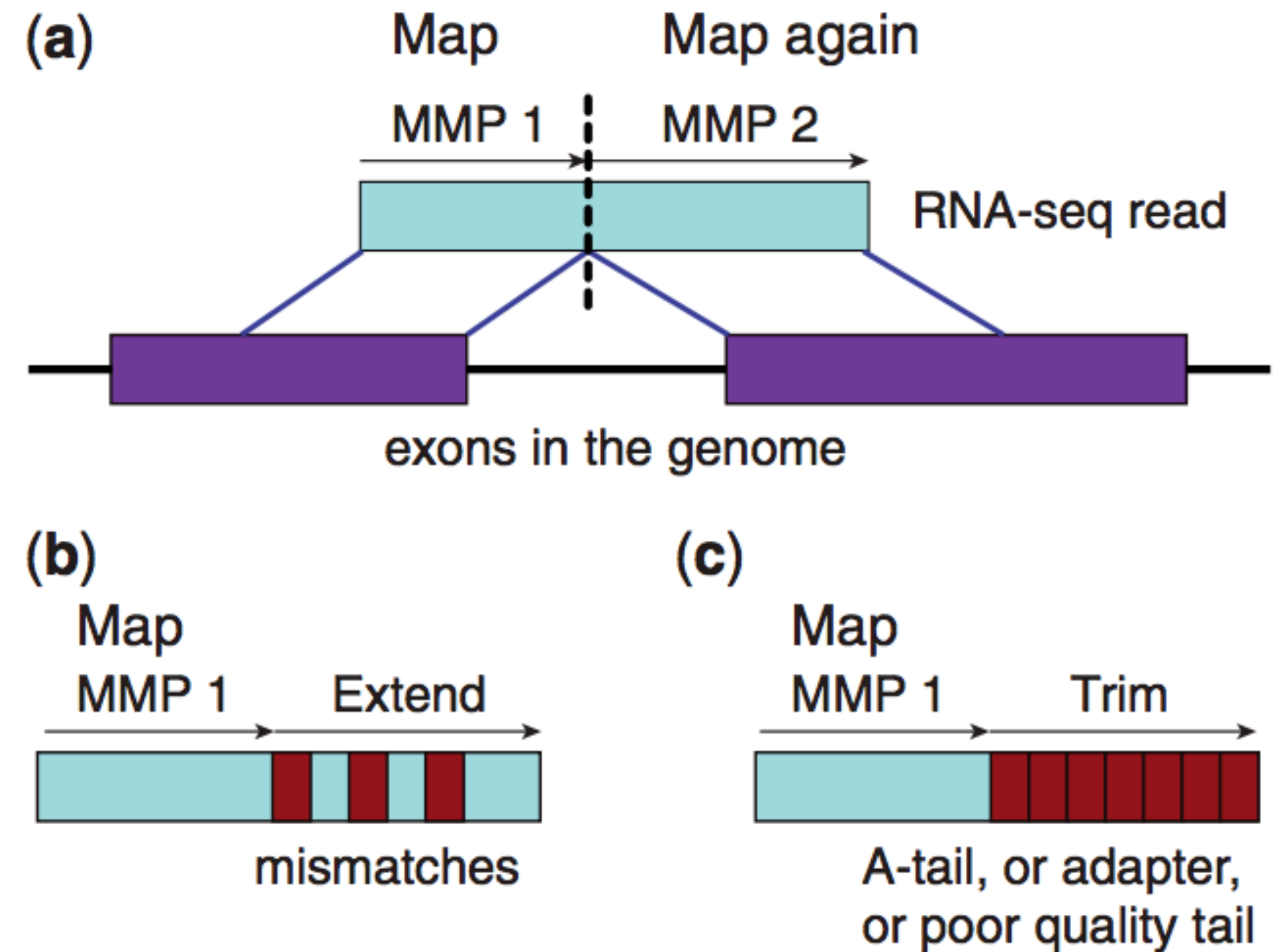
## Example 2: STAR Aligner

**Rationale:** Mapping of split reads is computationally slow.

**Solution:**

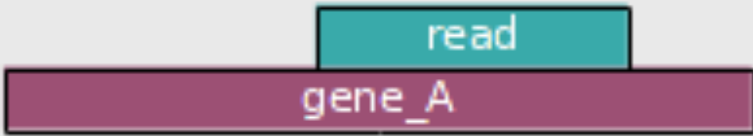
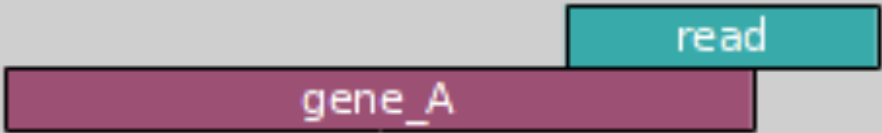
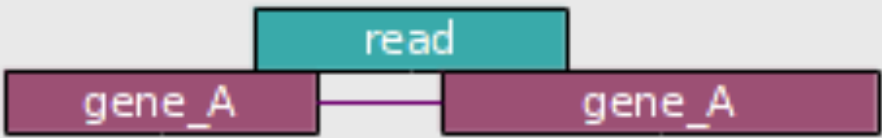

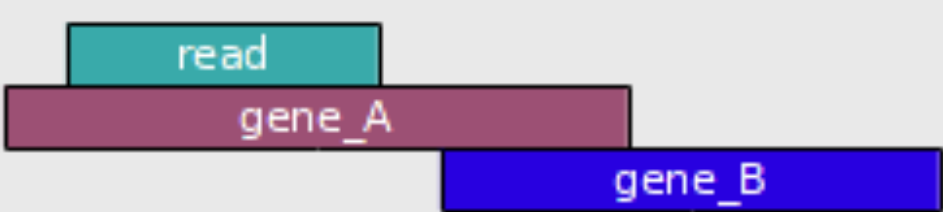
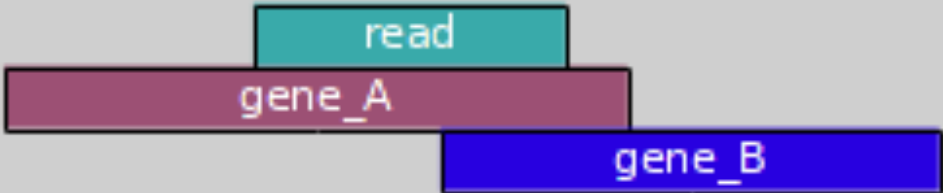
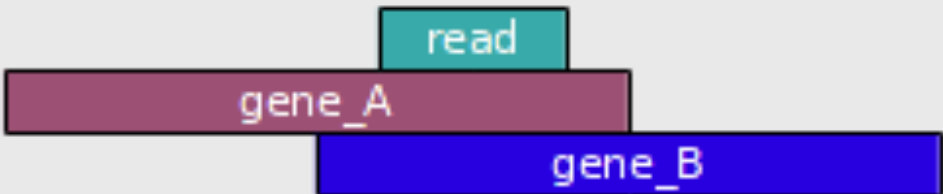
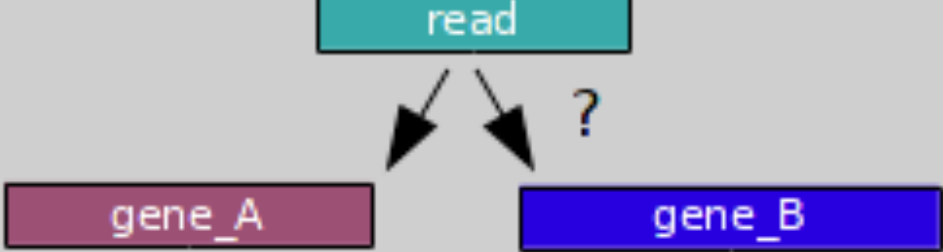
1. Use K-mer index
2. look for maximal mappable prefix/maximum matching portion (MMP) by extension
3. Split only when extension is not possible
4. Pieces back the split reads

**Results:** Much faster and more sensitive algorithm in detecting transcript

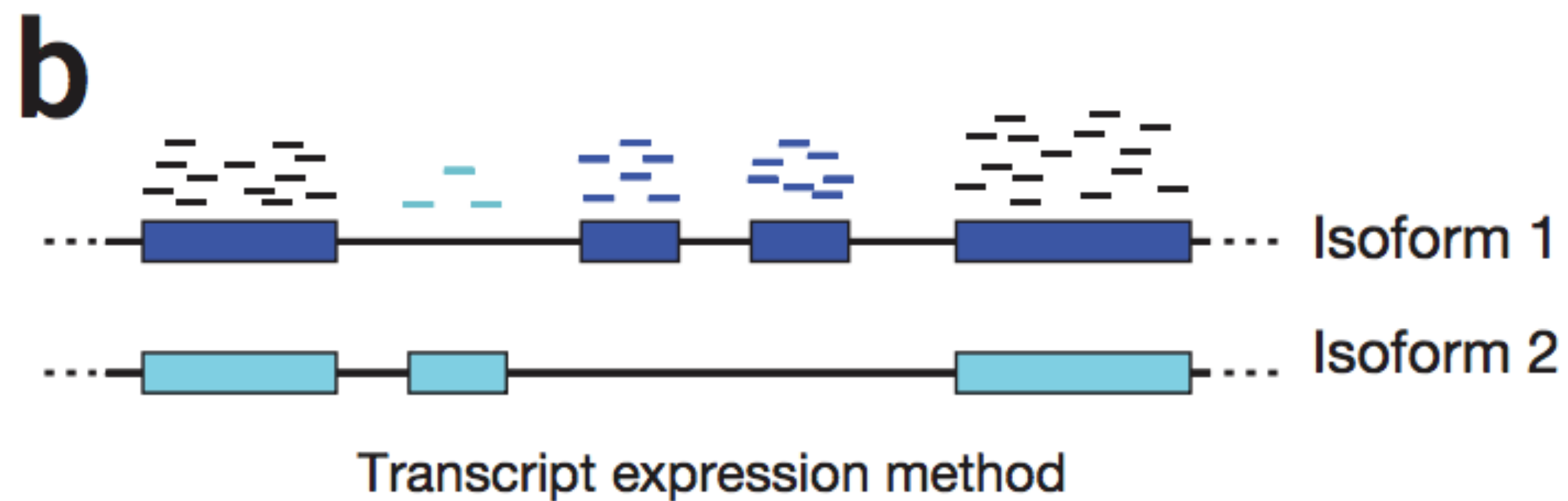
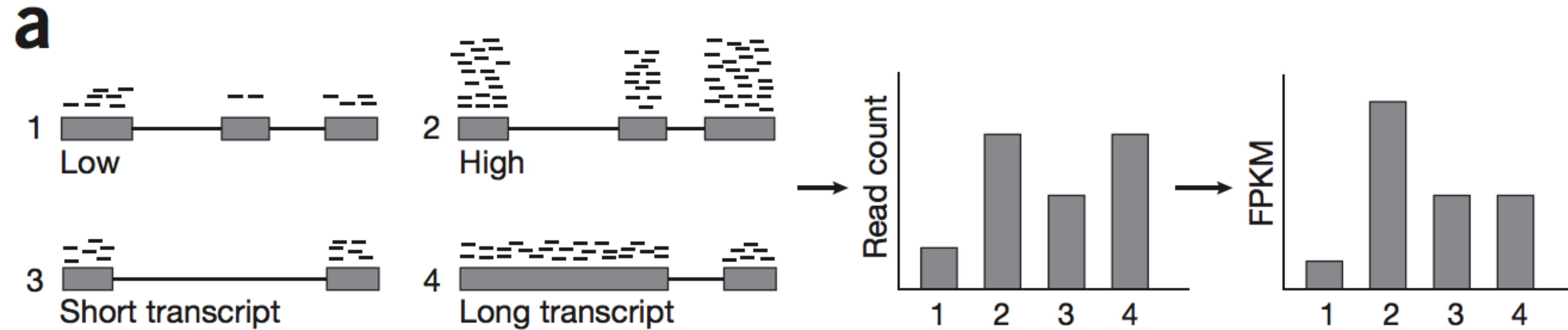


# Counting Mapped Reads by HTSeq

- Operate on aligned BAM files
- Total number of reads aligned to a gene is used as surrogate for gene expression level (called ‘raw read counts’)
- However, how reads are being counted is very much user defined !!!

	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 (both genes with --nonunique all)	gene_A	gene_A
	ambiguous (both genes with --nonunique all)		
	alignment_not_unique (both genes with --nonunique all)		

# Quantify Gene and Isoform Expression from Alignment



# Different philosophies of transcript quantification

---

- **Alignment-based:** Sequence alignment followed by counting of reads overlapping with a given annotated gene
- **Pseudo-Alignment:** Sometimes called 'Alignment Free', Quantify the number of reads that are consistent with a given transcript ( the exact location within the transcript is ignored)



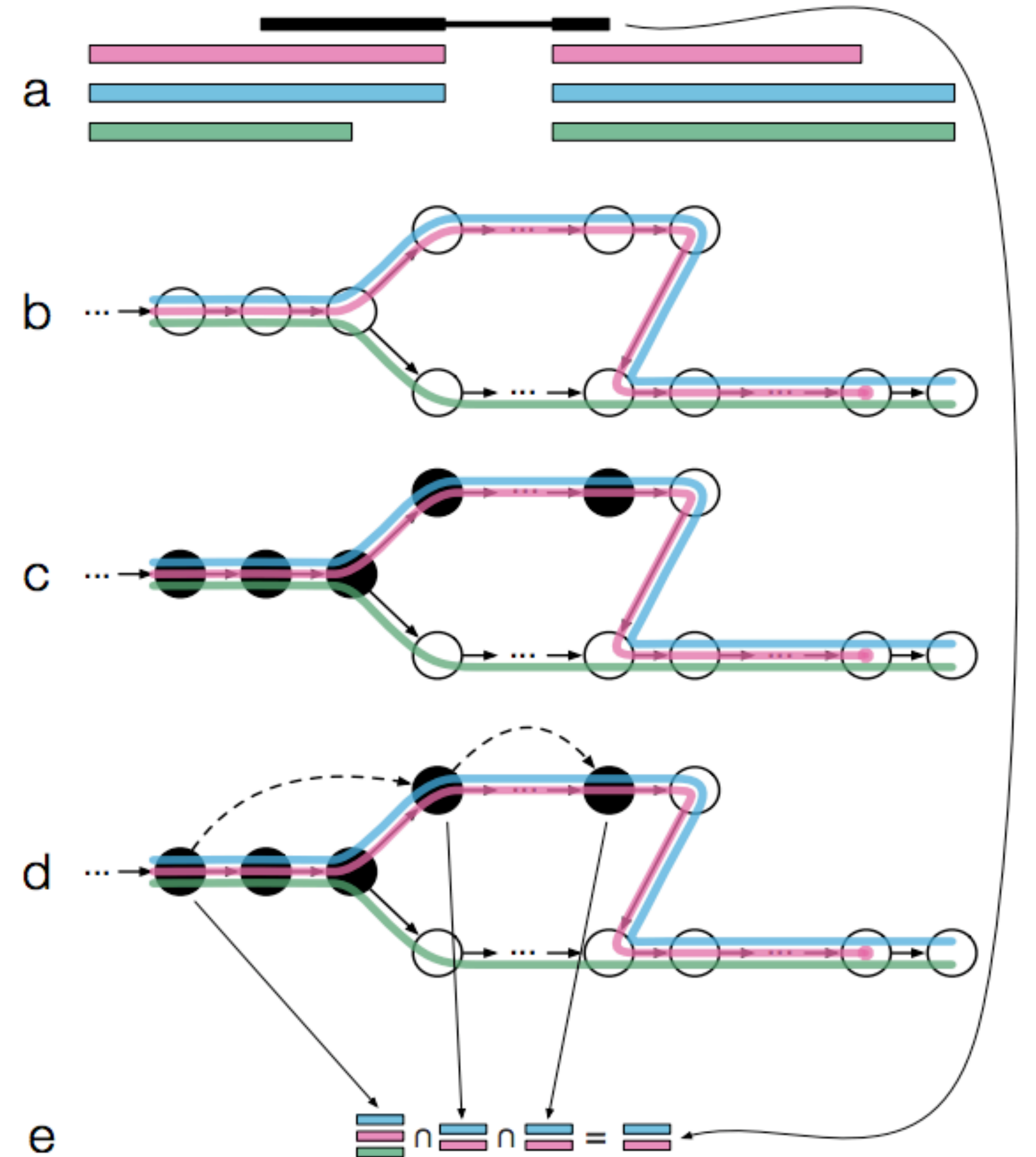
# Example: Kallisto

## Rationale:

1. Alignment based transcript reconstruction is computationally expensive
2. Read counting method introduces ambiguity

## Kallisto Algorithm:

1. Construct a graphical model (de Bruijn) for all known transcript isoforms
2. Allocate reads onto the graph model
3. Use Expectation-Maximization (EM) algorithm to estimate the number of transcripts



# Summary

---

## Alignment-Based

### Pros:

- Generate BAM files can be used for extensive QC and visualization

### Cons:

- Computationally expensive
- Maybe less accurate for transcript quantification

## Pseudo-Alignment

### Pros:

- Very fast and likely more accurate in most cases
- Computationally cheaper

### Cons:

- Does not generate BAM file
- Cannot visualize on IGV
- Less robust with low quality sequence data

# Quality Control of RNA-Seq Data

---

How many reads were aligned?

What were reasons for lack of alignment?

- Too many adaptor dimers?
- Too many rRNA reads?
- Bacterial Contamination?
- Too many repeated dimer?
- etc....

Do you have enough paired mates (for PE sequencing)?

Visual Inspection by  
IGV!

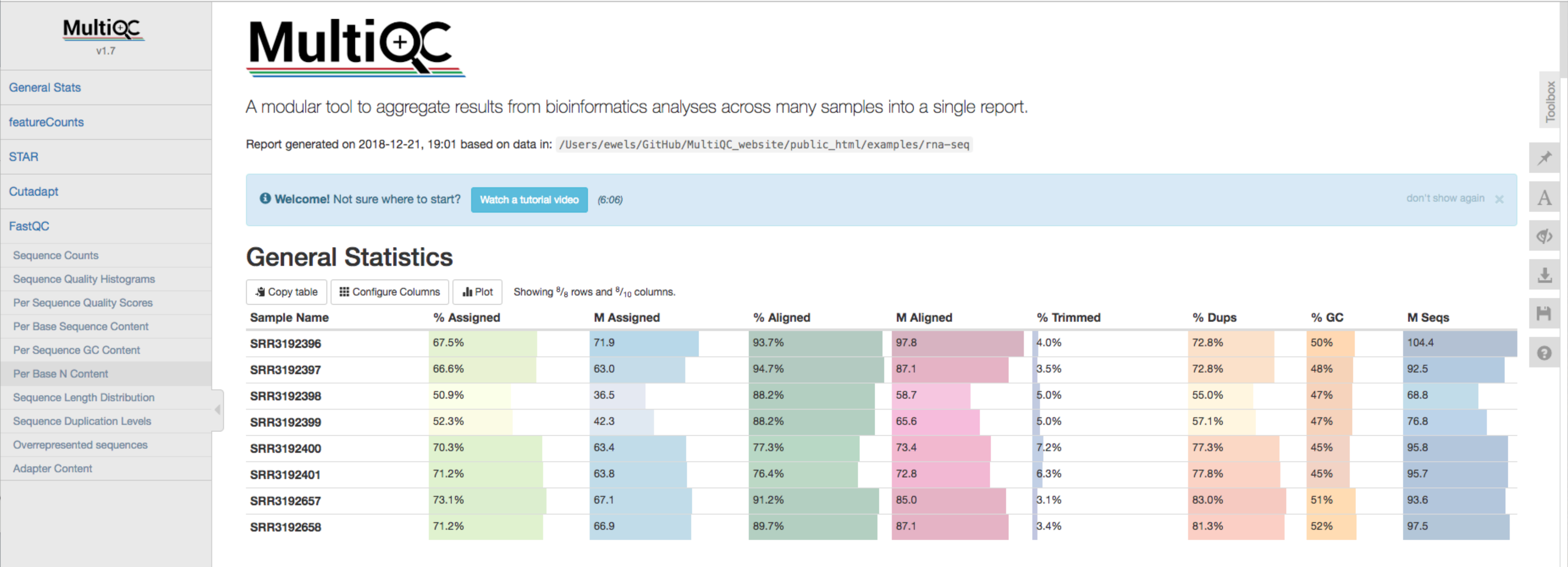
Use samtools, log files,  
QoRTs and visualize by  
MultiQC

# Example of IGV





# Example of MultiQC



[https://multiqc.info/examples/rna-seq/multiqc\\_report.html](https://multiqc.info/examples/rna-seq/multiqc_report.html)

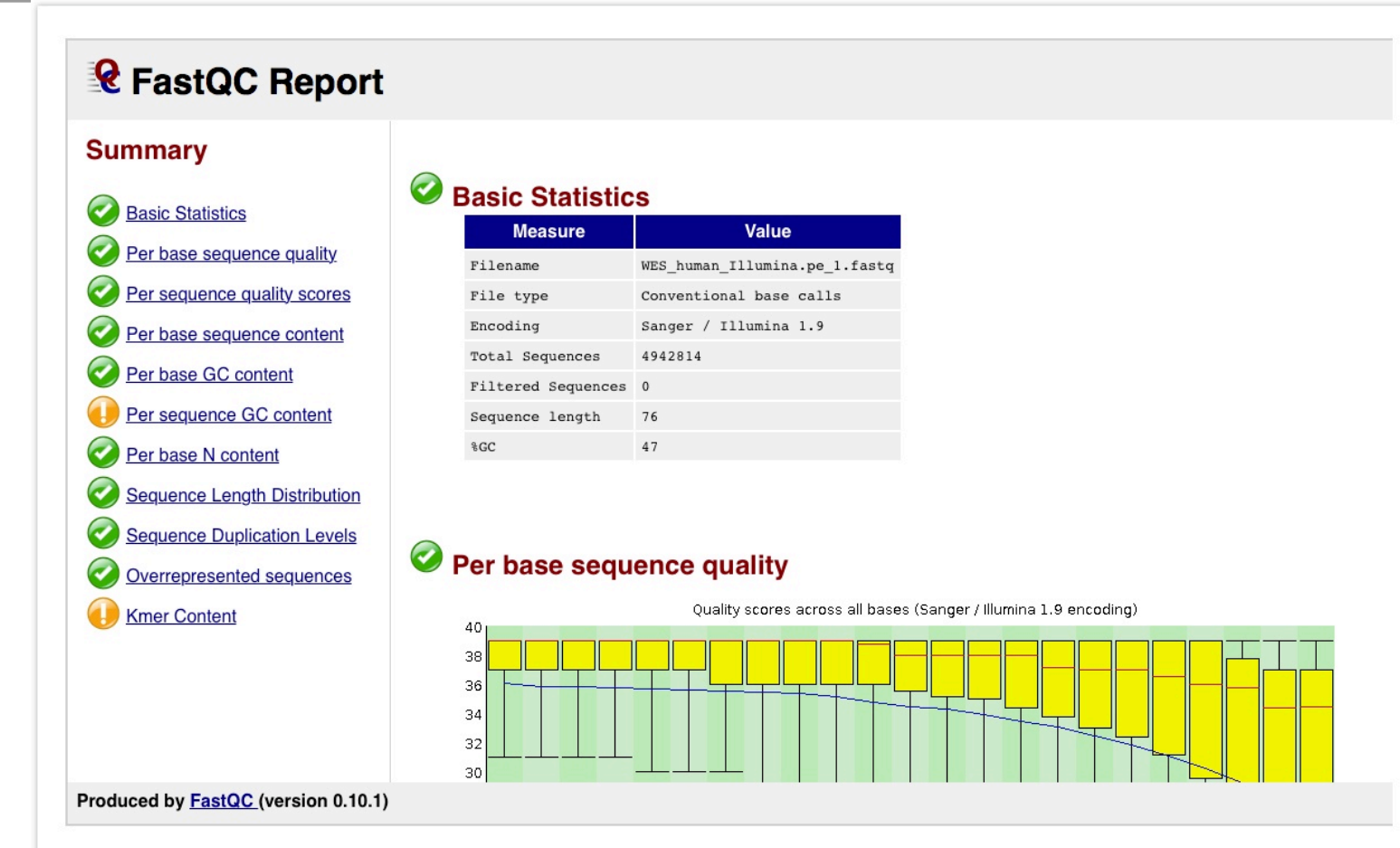
# Alignment Quality Control Summary

## raw reads QC

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

## aligned reads QC

- % (uniquely) aligned reads
- % exonic vs. intronic/intergenic
- gene diversity
- gene body coverage



FastQC

aligner's log files  
samtools flagstat  
RSeQC  
QoRTs  
MultiQC

# Normalize Sequencing Read for Expression Comparison

---

Reduce the impact of sequencing experiment variation and biases for more accurate comparison



# Gene Quantification and Normalization

**Raw counts:** number of reads (or fragments) overlapping with the union of exons of a gene



**Raw Count  $\neq$  Expression level**

**Raw Count is strongly influenced by:**

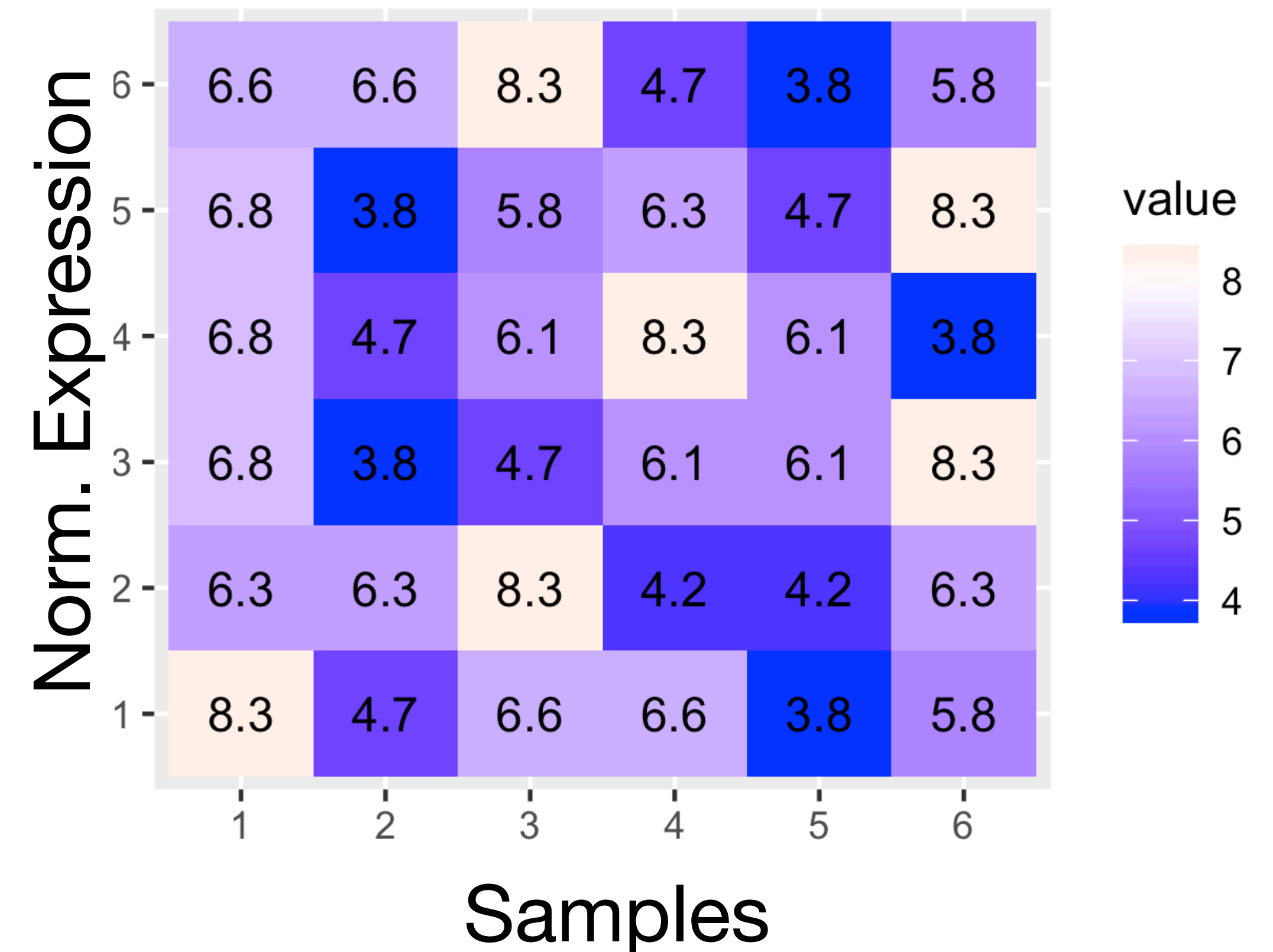
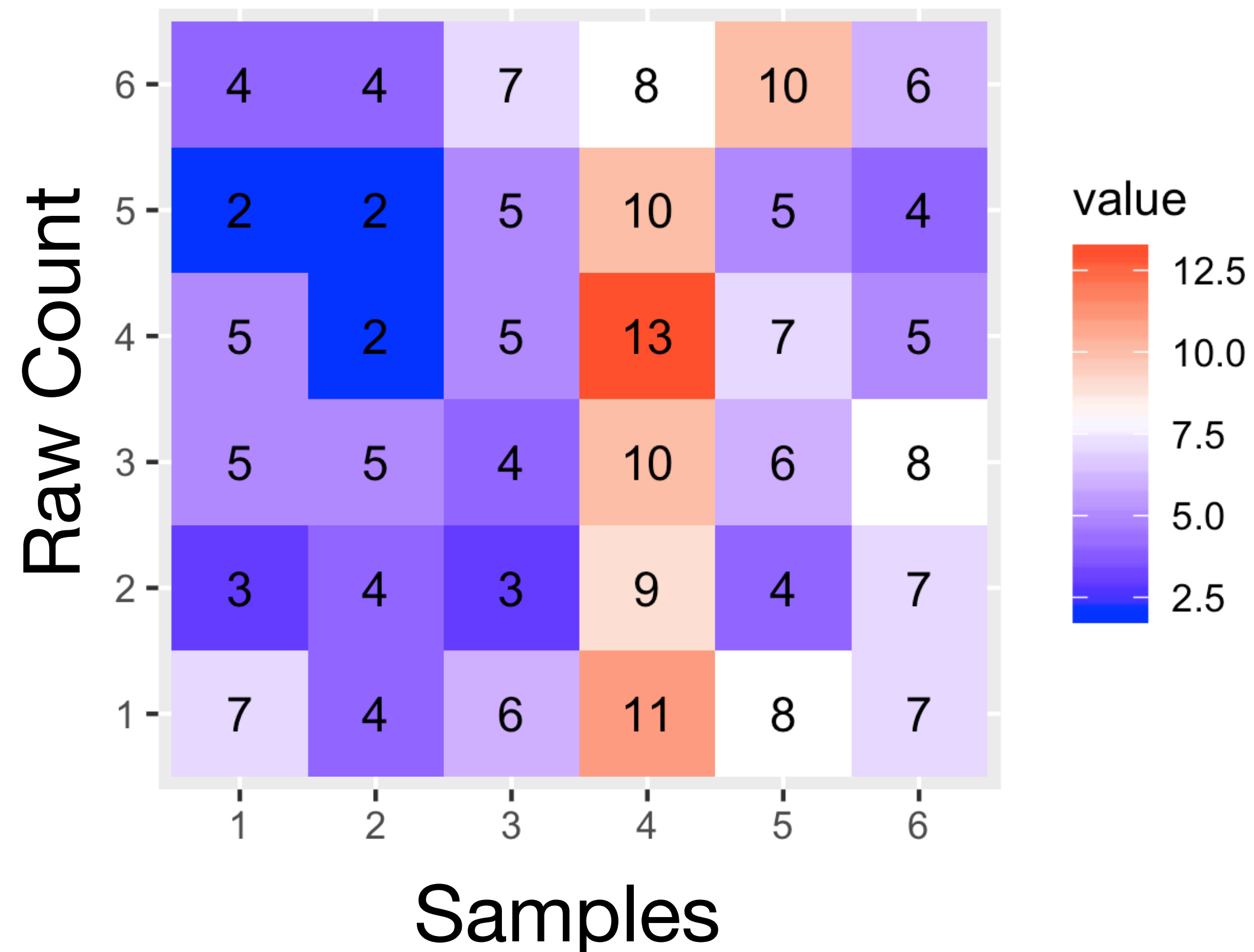
- gene length
- transcript sequence (% GC)
- sequencing depth
- expression of all other genes in the same sample

may cause variations for different genes expressed at the same level

may cause variations for same genes in different sample

# How Normalization Supposed to Work

Normalization attempts to account for experimental biases and noises to put sample expression on the **same scale** for comparison



# Common Normalized Measurements

---

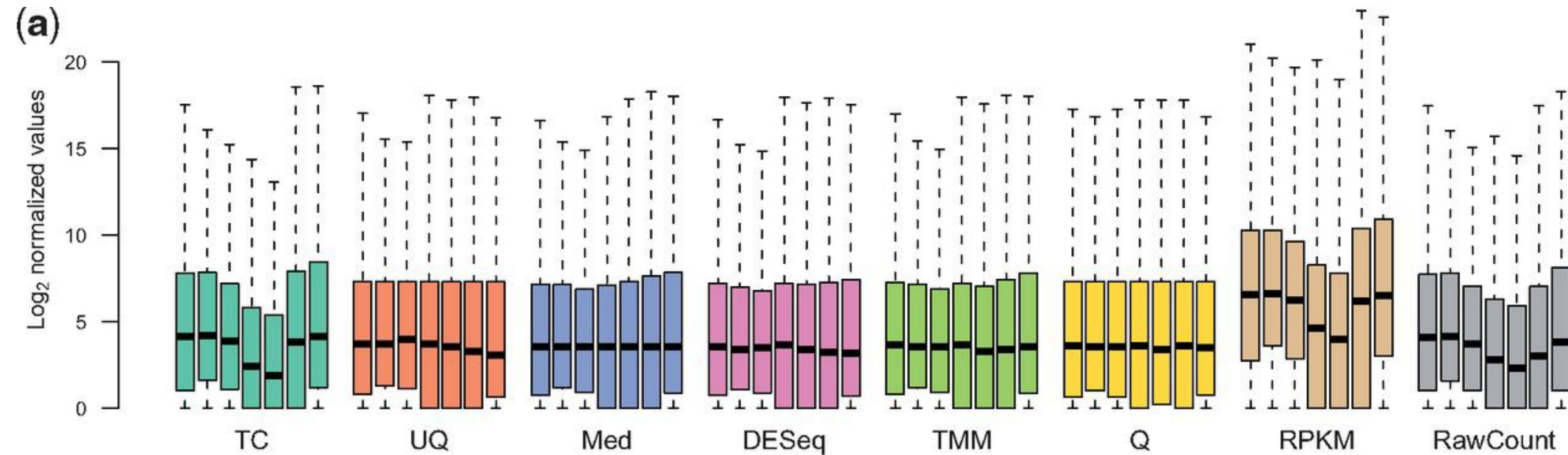
- **Raw Counts:** number of reads/fragments overlapping all the exons of a gene
- **RPKM/FPKM:** Reads/fragments per kilobase of gene per million reads mapped
- **TPM:** transcripts per million reads mapped=[gene / read count per bp/all gene count per all gene bp]
- **rlog:** log2-transformed count data normalized for small counts and library size. There are many variations:
  - Trimmed mean of M values (TMM)
  - DESeq2
  - Upper-Quartile (UQ)

$$X_i$$

$$RPKM_i = \frac{X_i}{\left(\frac{l_i}{10^3}\right)\left(\frac{N}{10^6}\right)}$$

$$TPM_i = \frac{X_i}{l_i} * \frac{10^6}{\sum \frac{X_j}{l_k}}$$

# Effects of normalization methods on FC calculation and DGE analysis



- All normalization method makes assumptions on the property of the data
- [RF]PKM assumes even distribution of reads across your transcript of interest... **NOT TRUE!**
- Avoid [RF]PKM normalization for DGE!
- If you need normalized expression values, use TPM or DESeq's rlog