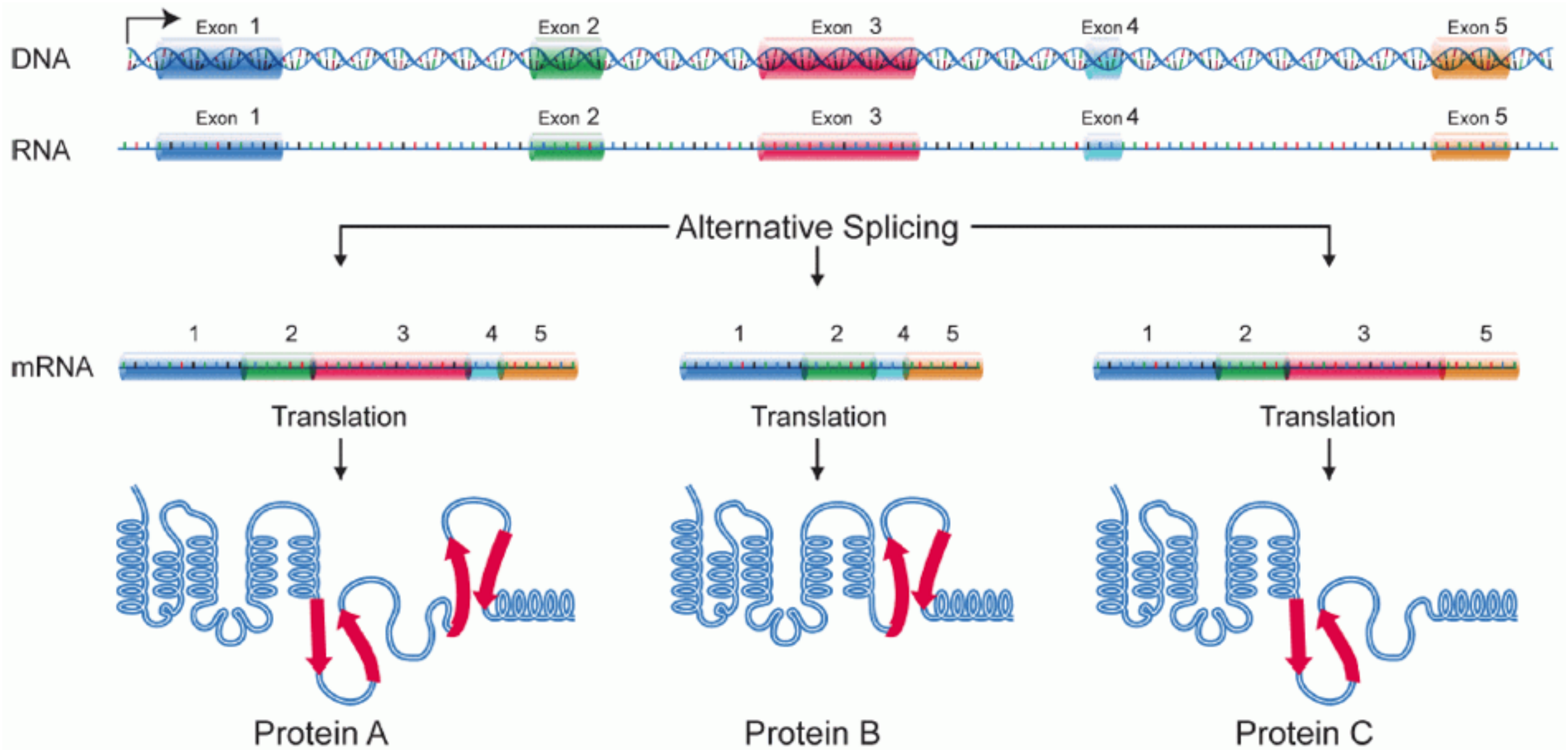
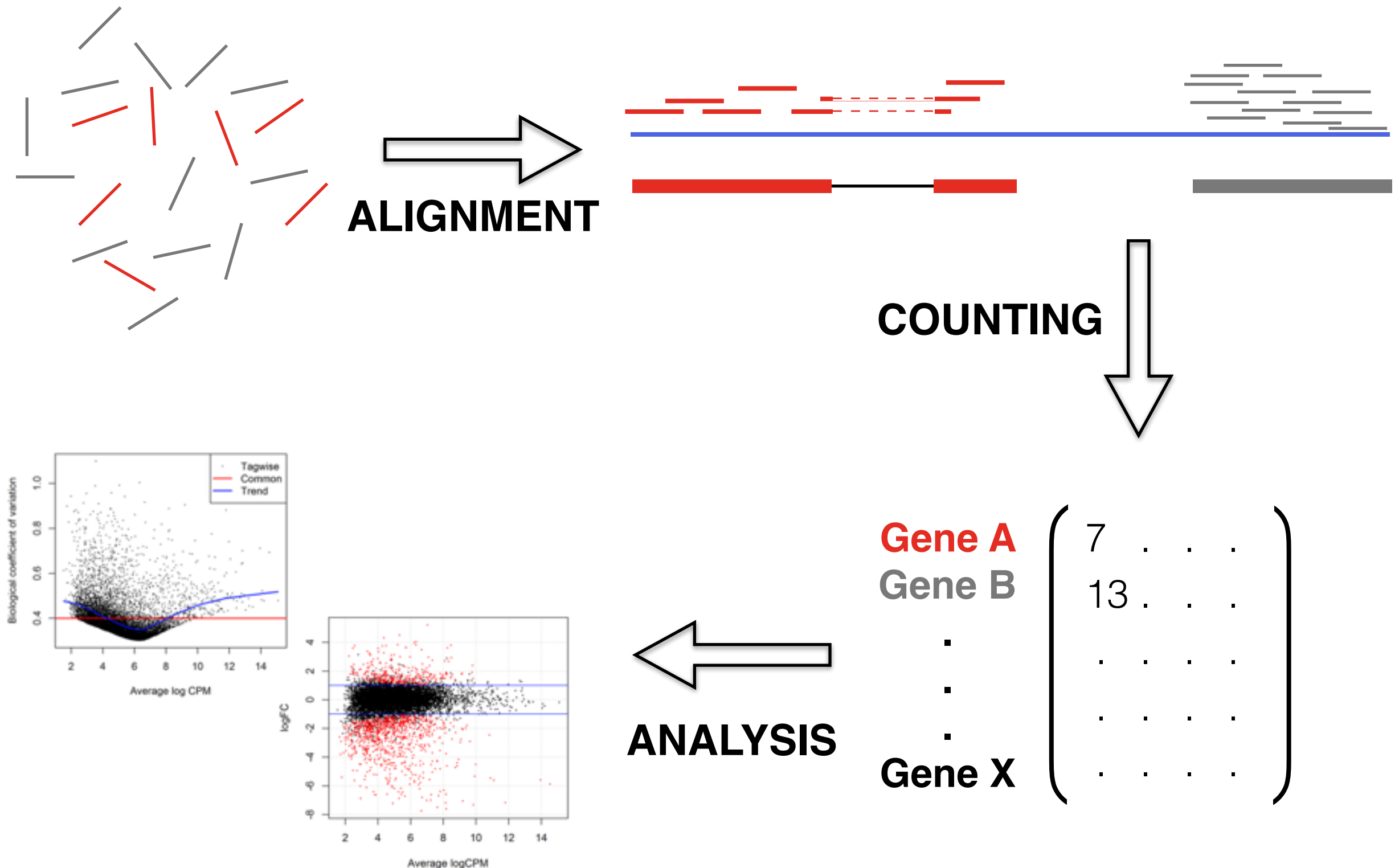


Modern RNA-seq analysis

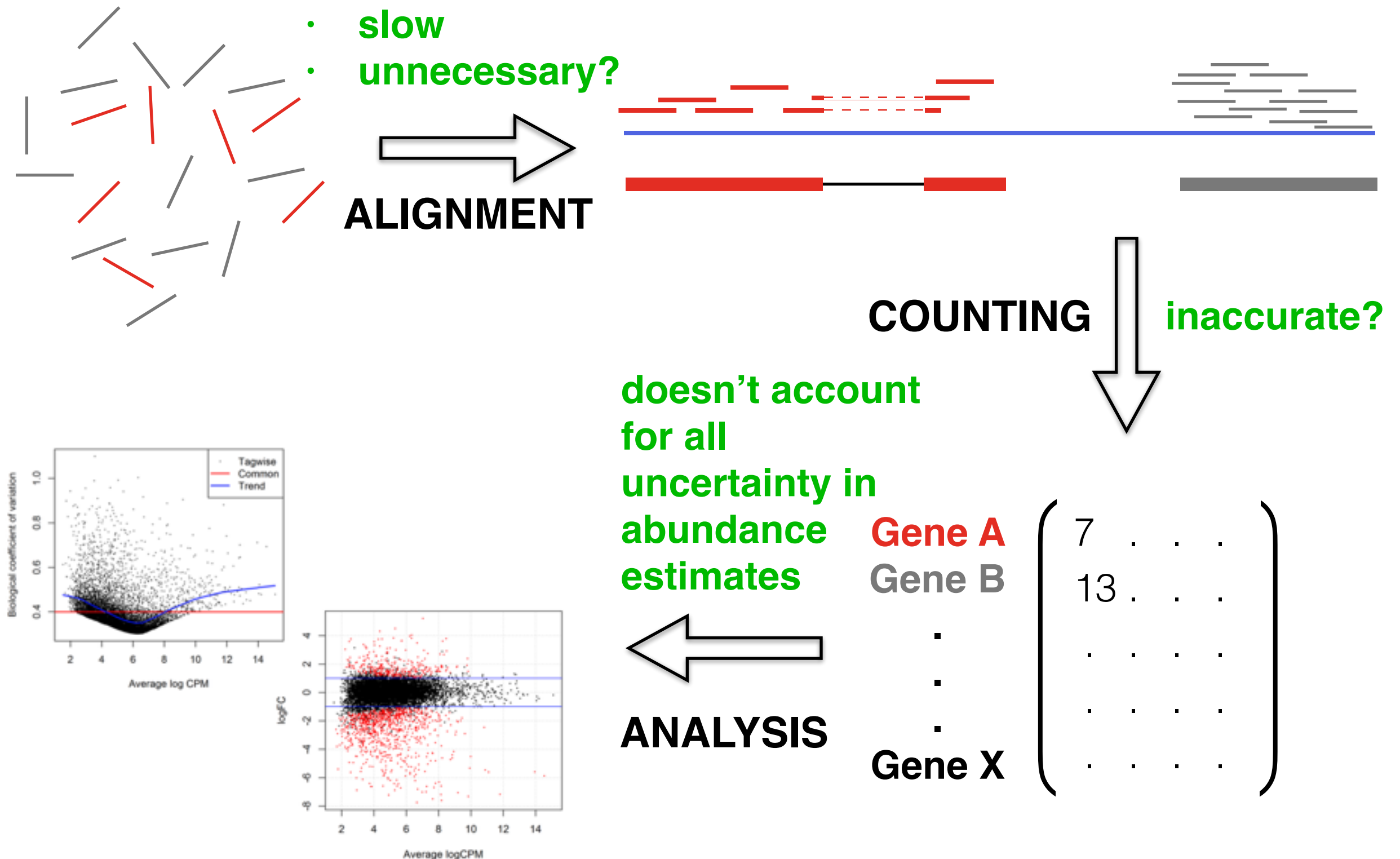
Charlotte Soneson
University of Zurich
Brixen 2016



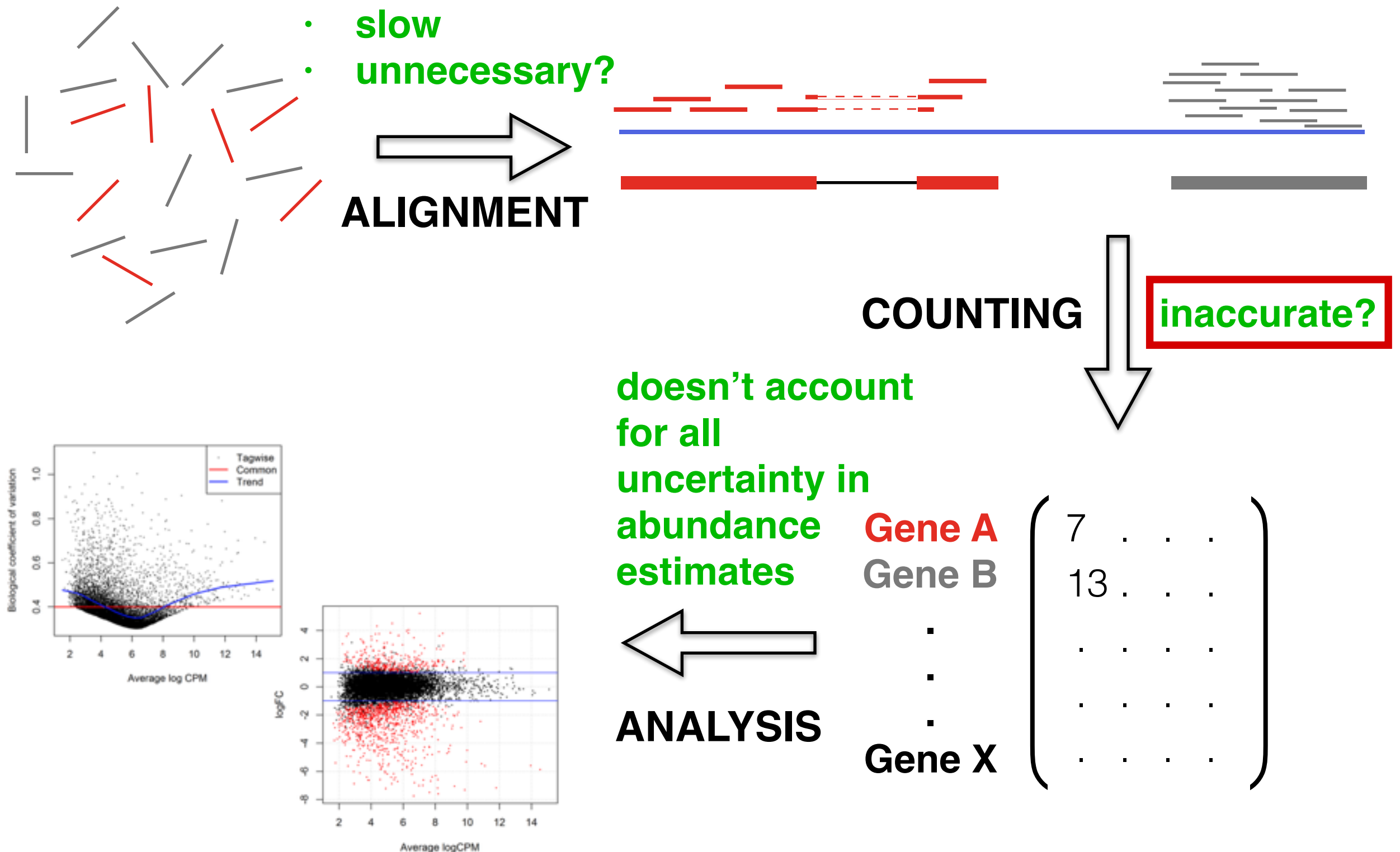
The traditional workflow



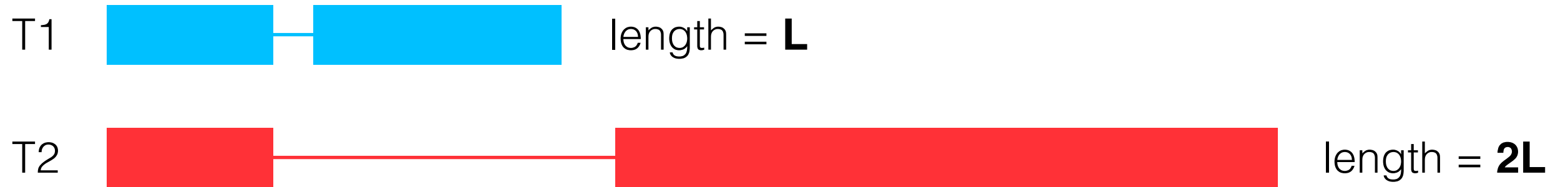
The traditional workflow



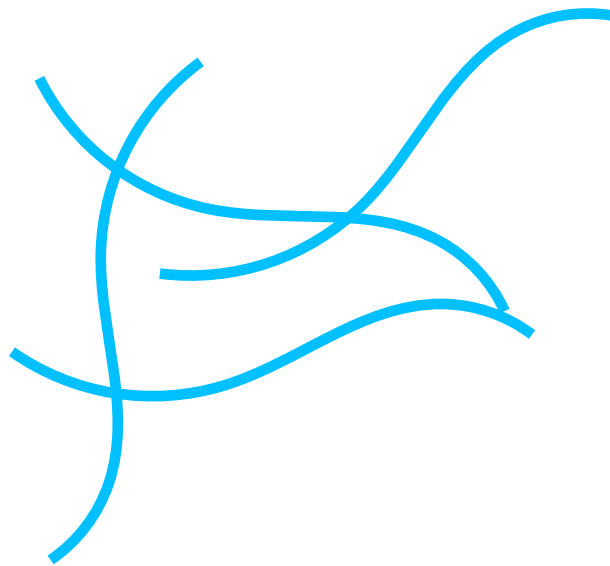
The traditional workflow



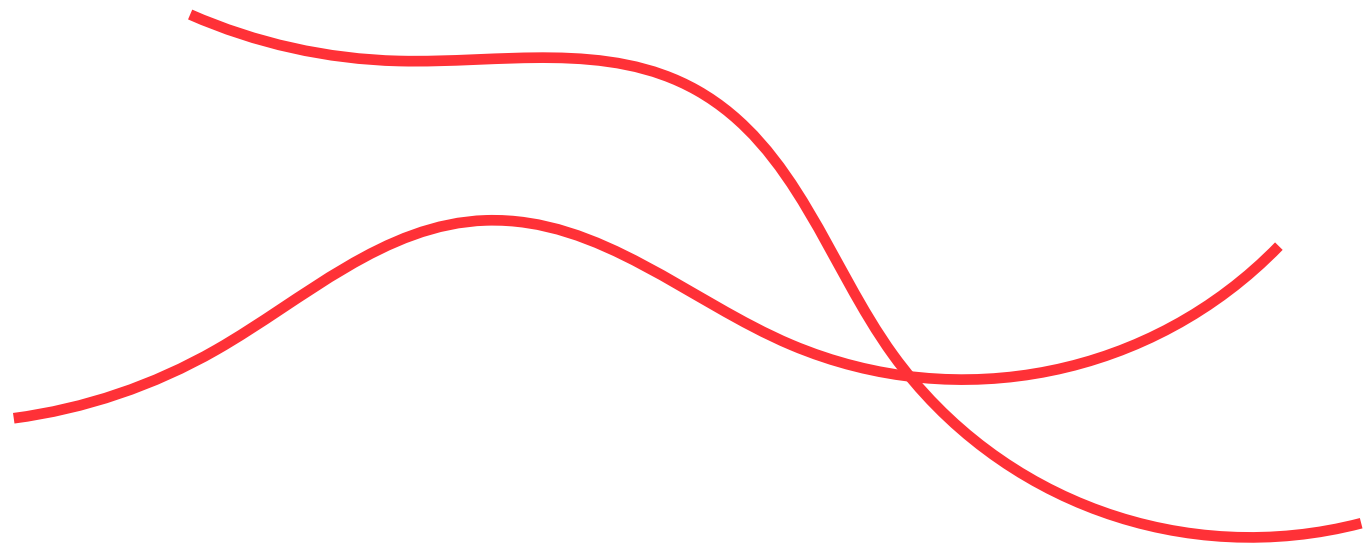
Abundance quantification



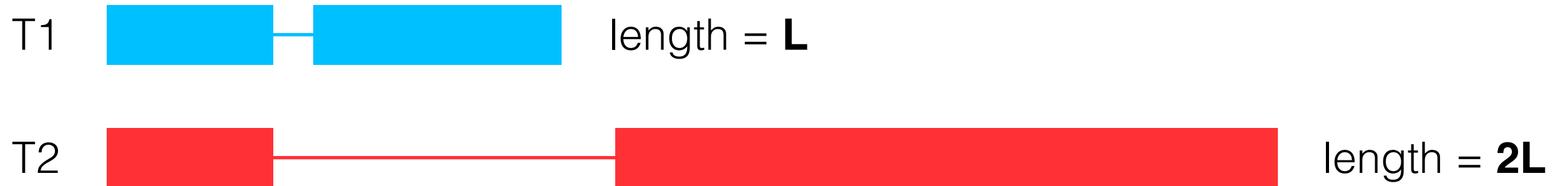
sample 1



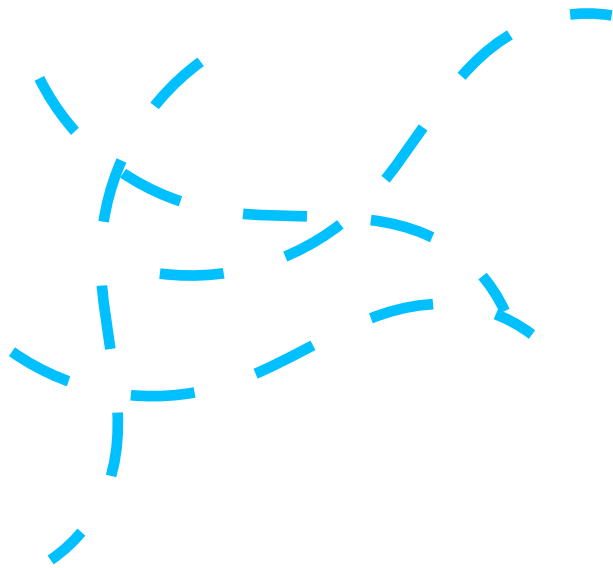
sample 2



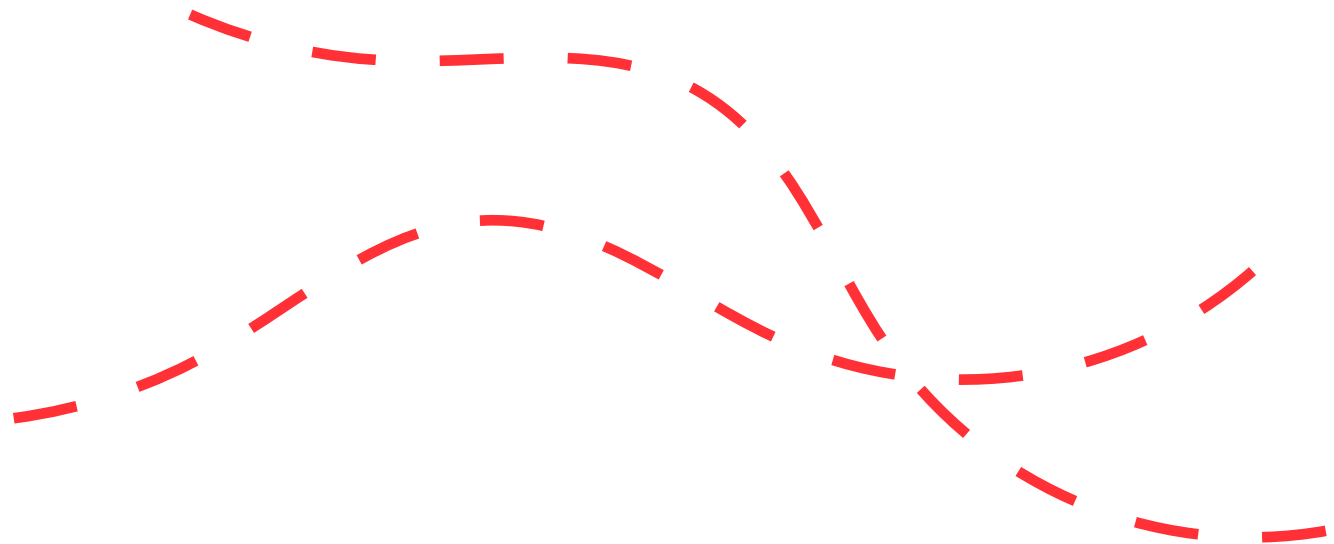
Abundance quantification



sample 1



sample 2



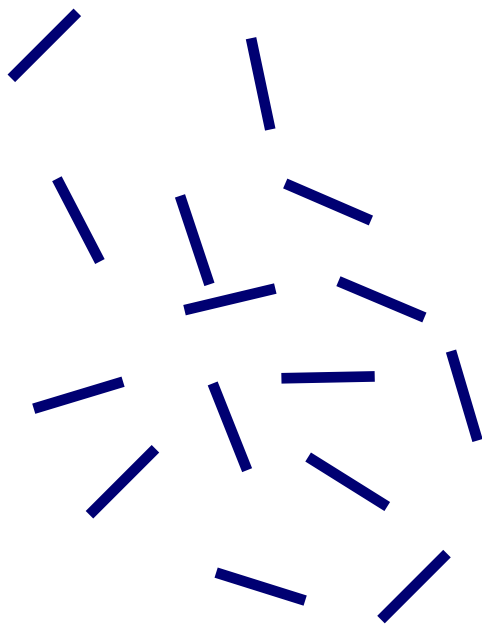
Gene-level read counts

T1  length = L

T2  length = $2L$

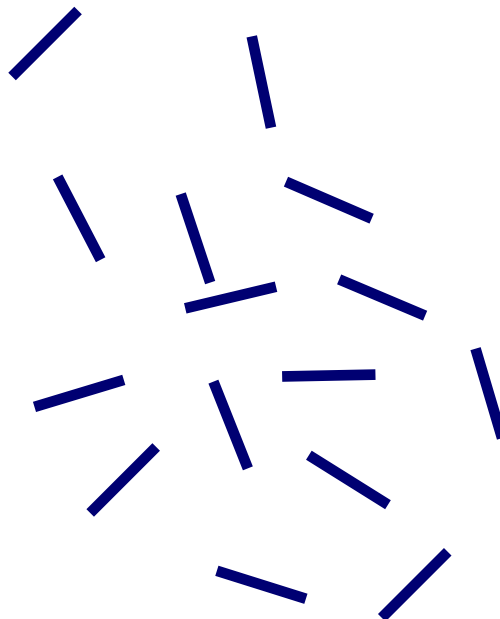
Gene  length = $2.6L$

sample 1



150 reads

sample 2



150 reads

Gene	S1	S2
Count	150	150

What can we do?

- Consider another abundance unit that better reflects the underlying abundances (“number of transcript molecules”)
- Include “adjustment” of gene counts to reflect underlying isoform composition

What can we do?

How can we get such values?

Are they any good?

- Consider another abundance unit that better reflects the underlying abundances (“number of transcript molecules”)
- Include “adjustment” of gene counts to reflect underlying isoform composition

How could such adjustment be done?

What can we do?

How can we get such values?

good?

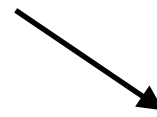
We need transcript-level information!

- Consider another approach that reflects the underlying number of transcripts
- Incorporate gene counts to reflect underlying composition

How could such adjustment be done?

Abundance units

read count for transcript i



C_i



l_i



length of transcript i

Abundance units

read count for transcript i

C_i



l_i

fragment
length

length of transcript i

$$t_i = \frac{C_i r}{l_i}$$

Abundance units

read count for transcript i

c_i



l_i

fragment
length

length of transcript i

$$t_i = \frac{c_i r}{l_i}$$

$$TPM_i = 10^6 \cdot \frac{t_i}{\sum_k t_k}$$

Abundance units

read count for transcript i

c_i



fragment
length

$$t_i = \frac{c_i r}{\ell_i}$$

length of transcript i

$$TPM_i = 10^6 \cdot \frac{t_i}{\sum_k t_k}$$

library size

$$RPKM_i = 10^9 \cdot \frac{c_i}{\ell_i \sum_k c_k} = 10^9 \cdot \frac{t_i}{\sum_k (t_k \ell_k)}$$

Abundance units

read count for transcript i

c_i



ℓ_i

fragment
length

length of transcript i

$$t_i = \frac{c_i r}{\ell_i}$$

$$TPM_i = 10^6 \cdot \frac{t_i}{\sum_k t_k}$$

library size

$$RPKM_i = 10^9 \cdot \frac{c_i}{\ell_i \sum_k c_k} = 10^9 \cdot \frac{t_i}{\sum_k (t_k \ell_k)}$$

$$TPM_i \propto RPKM_i$$

$$\sum_i TPM_i = 10^6$$

Offsets (“average transcript lengths”)


- Similar to correction factors for library size, but sample-**and** gene-specific
- Weighted average of transcript lengths, weighted by estimated abundances (TPMs)
- Average transcript length for gene g in sample s :

$$ATL_{gs} = \sum_{i \in g} \theta_{is} \bar{\ell}_{is}, \quad \sum_{i \in g} \theta_{is} = 1$$

$\bar{\ell}_{is}$ = effective length of isoform i (in sample s)

θ_{is} = relative abundance of isoform i in sample s

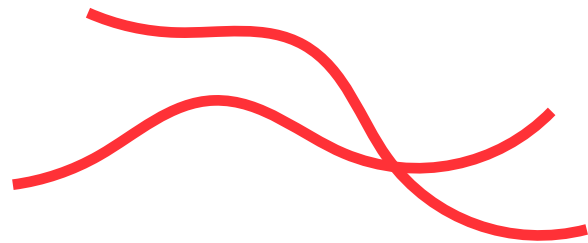
Average transcript lengths

T1  length = **L**

T2  length = **2L**




$$ATL_{g1} = 1 \cdot L + 0 \cdot 2L = L$$

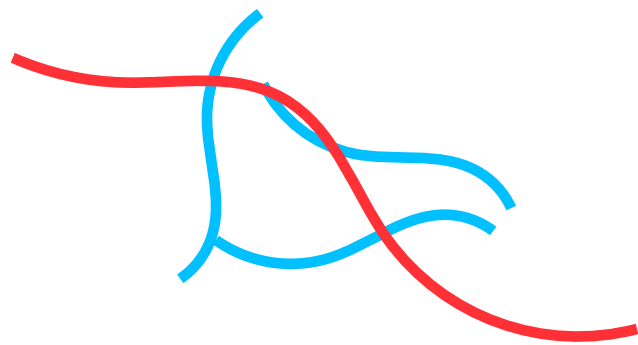


$$ATL_{g2} = 0 \cdot L + 1 \cdot 2L = 2L$$

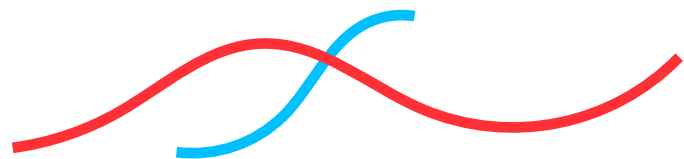
Average transcript lengths

T1  length = **L**

T2  length = **2L**

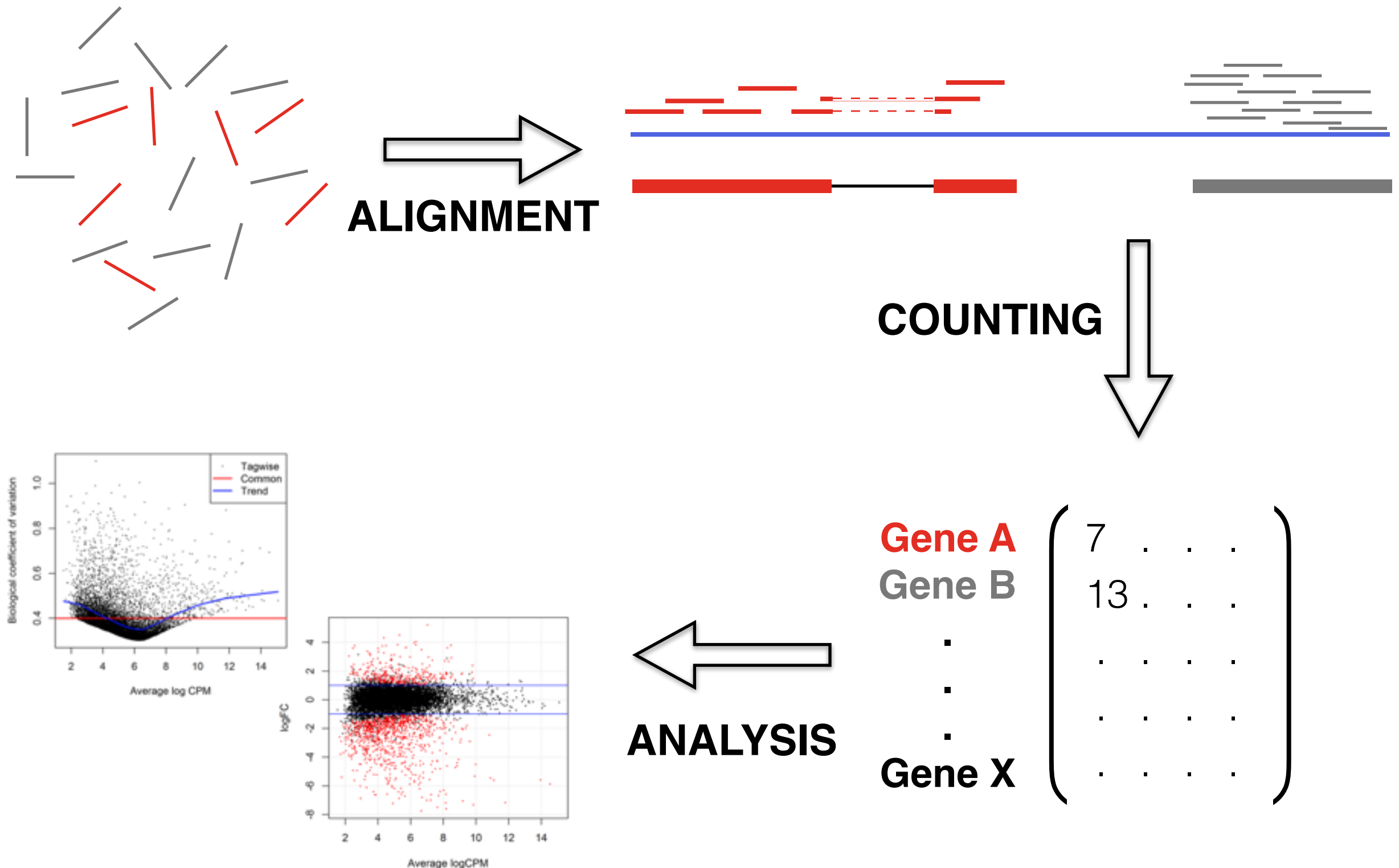


$$ATL_{g1} = 0.75 \cdot L + 0.25 \cdot 2L = 1.25L$$

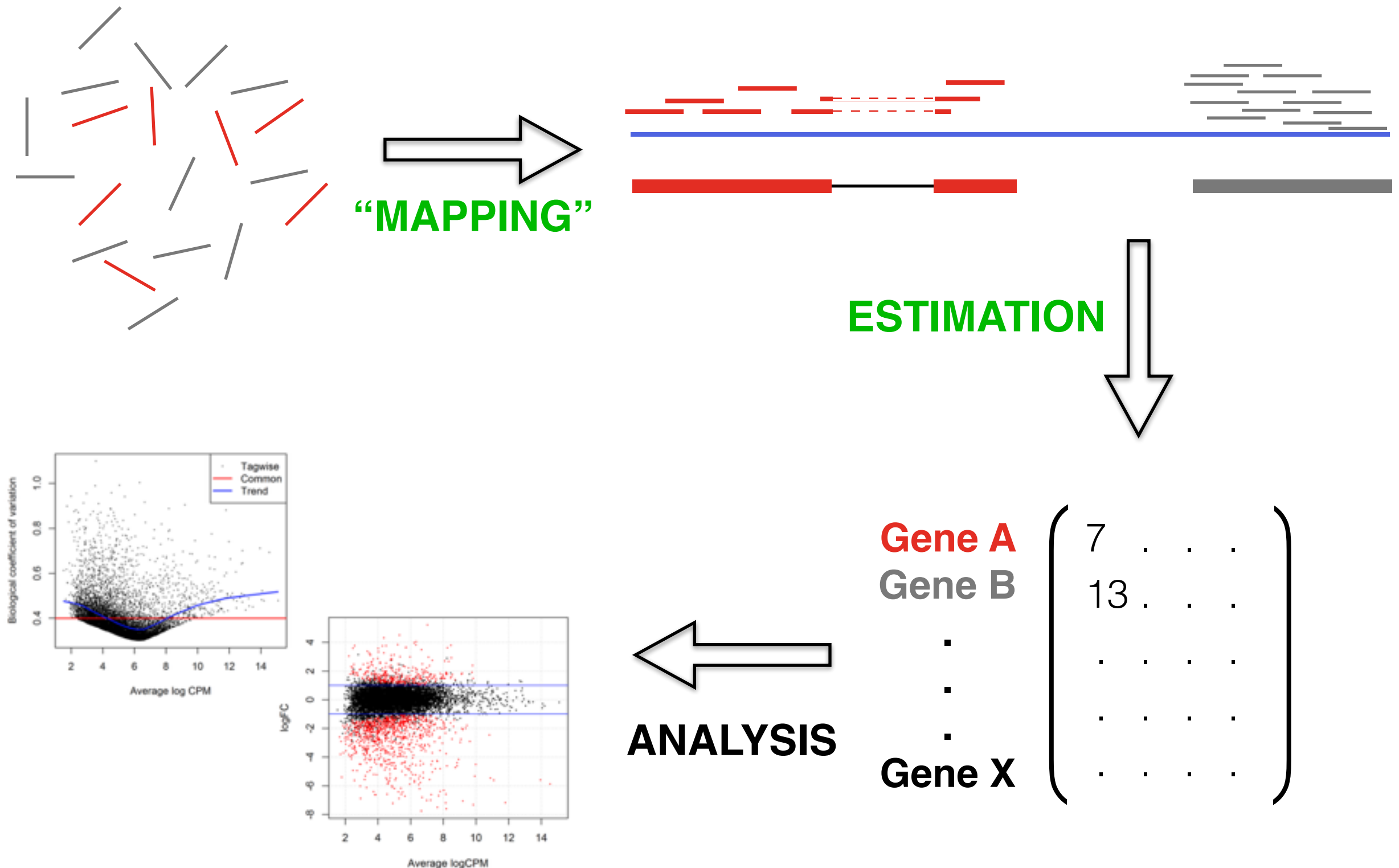


$$ATL_{g2} = 0.5 \cdot L + 0.5 \cdot 2L = 1.5L$$

The traditional workflow



The “modern” workflow



The “mapping” step

- Does not provide “full” alignment information (i.e., no exact base-by-base alignment).
- Rather, finds all transcripts (and positions) that a read is compatible with.
- Comes in various flavors:
 - pseudoalignment (kallisto)
 - lightweight alignment (Salmon)
 - quasimapping (Sailfish, RapMap)

The “estimation” step

- Input: for each read, the “equivalence class” of compatible transcripts
- Probabilistic modeling of read generation process, with transcript abundance as parameter
- EM algorithm
- Output: estimated abundance of each transcript

Step 1: build transcriptome index

kallisto

name of index



```
vpn-89-206-119-17:~ charlotte$ kallisto index -i transcriptome_index  
.idx transcripts.fasta
```



transcriptome fasta file

Salmon

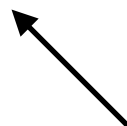
transcriptome fasta file



```
vpn-89-206-119-17:~ charlotte$ salmon index -t transcripts.fasta -i  
transcriptome_index.idx -p 10
```



name of index



number of cores

Where to find transcript fasta?

www.ensembl.org/info/data/ftp/index.html

Single species data

Popular species are listed first. You can customise this list via our [home page](#).

Show 10 entries											Show/hide columns
★	Species	DNA (FASTA)	cDNA (FASTA)	CDS (FASTA)	ncRNA (FASTA)	Protein sequence (FASTA)	Annotated sequence (EMBL)	Annotated sequence (GenBank)	Gene sets	Whole databases	Variation (GV)
Y	Human <i>Homo sapiens</i>	FASTA	FASTA	FASTA	FASTA	FASTA	EMBL	GenBank	GTF GFF3	MySQL	GV
Y	Mouse <i>Mus musculus</i>	FASTA	FASTA	FASTA	FASTA	FASTA	EMBL	GenBank	GTF GFF3	MySQL	GV
Y	Zebrafish <i>Danio rerio</i>	FASTA	FASTA	FASTA	FASTA	FASTA	EMBL	GenBank	GTF GFF3	MySQL	GV

Where to find transcript fasta?

www.ensembl.org/info/data/ftp/index.html

reference files for alignment-based workflow

Single species data

Popular species are listed first. You can customise this list via our [home page](#).

Show 10 entries Show/hide columns										
★	Species	DNA (FASTA)	cDNA (FASTA)	CDS (FASTA)	ncRNA (FASTA)	Protein sequence (FASTA)	Annotated sequence (EMBL)	Annotated sequence (GenBank)	Gene sets	Whole databases
Y	Human <i>Homo sapiens</i>	FASTA	FASTA	FASTA	FASTA	FASTA	EMBL	GenBank	GTF GFF3	MySQL
Y	Mouse <i>Mus musculus</i>	FASTA	FASTA	FASTA	FASTA	FASTA	EMBL	GenBank	GTF GFF3	MySQL
Y	Zebrafish <i>Danio rerio</i>	FASTA	FASTA	FASTA	FASTA	FASTA	EMBL	GenBank	GTF GFF3	MySQL

Step 2: quantify

kallisto

name of index



```
vpn-89-206-119-17:~ charlotte$ kallisto quant -i transcriptome_index  
.idx -o results/sample1 -b 30 -t 10 sample1_1.fastq sample1_2.fastq
```



output folder

bootstraps



number of cores



input fastq files



Salmon

name of index



libtype



```
vpn-89-206-119-17:~ charlotte$ salmon quant -i transcriptome_index.idx -l IU  
-p 10 -1 sample1_1.fq -2 sample1_2.fq -o results/sample1 --numBootstraps 30
```



number
of cores



input fastq files



output folder

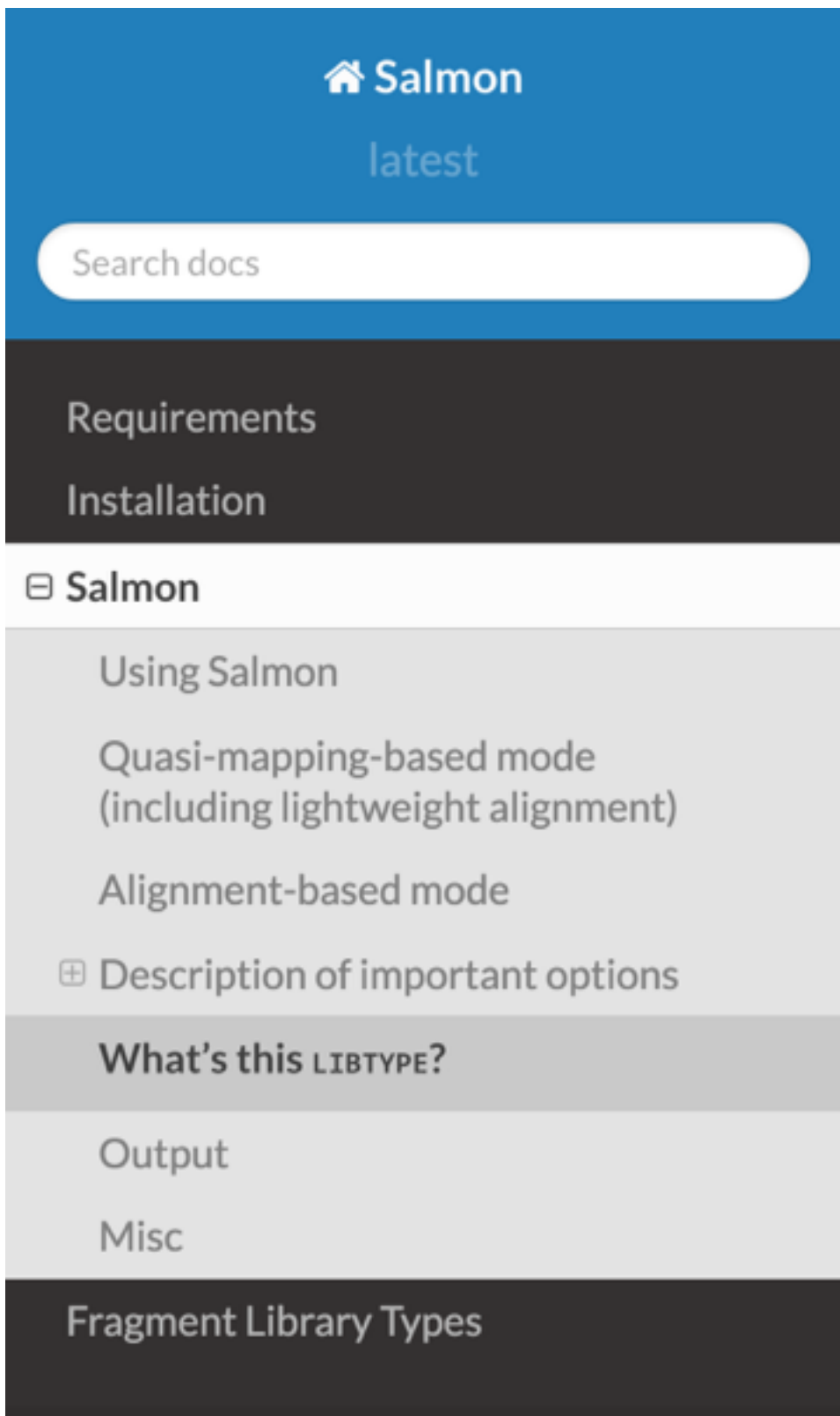


bootstraps



Salmon LIBTYPE argument

<http://salmon.readthedocs.io/en/latest/salmon.html#what-s-this-libtype>



What's this **LIBTYPE**?

Salmon, like sailfish, has the user provide a description of the type of sequencing library from which the reads come, and this contains information about e.g. the relative orientation of paired end reads. However, we've replaced the somewhat esoteric description of the library type with a simple set of strings; each of which represents a different type of read library. This new method of specifying the type of read library is being backported into Sailfish and will be available in the next release.

The library type string consists of three parts: the relative orientation of the reads, the strandedness of the library, and the directionality of the reads.

The first part of the library string (relative orientation) is only provided if the library is paired-end. The possible options are:



























```
I = inward
O = outward
M = matching
```

output

kallisto

	abundance.h5	
	abundance.tsv	
	run_info.json	

Salmon

	AHY9U3~9				bootstrap		
	cmd_info.json				expected_bias.gz		
	libFormatCounts.txt				fld.gz		
	libParams				meta_info.json		
	logs				observed_bias.gz		
	quant.sf						

output

kallisto

[abundance.tsv]

target_id	length	eff_length	est_counts	tpm
ENST00000406070	2025	1874.91	0	0
ENST00000446844	2227	2076.91	3.37465	0.129755
ENST00000599620	686	535.97	0	0
ENST00000471557	505	355.404	2.84168	0.638509
ENST00000338761	1456	1305.91	1.3122e-05	8.02414e-07
ENST00000417509	1444	1293.91	5.15988	0.318455
ENST00000484946	610	460.029	17.4159	3.02326
ENST00000490656	660	509.97	7.51996	1.17756
ENST00000439537	1161	1010.91	14.432	1.14006
ENST00000493251	641	491.006	2.63203	0.428073
ENST00000460127	408	259.526	0	0

Salmon

[quant.sf]

Name	Length	EffectiveLength	TPM	NumReads
ENST00000406070	2025	1869.81	0	0
ENST00000446844	2227	2071.81	0.137334	3.71695
ENST00000599620	686	530.936	0	0
ENST00000471557	505	350.256	0.731211	3.3457
ENST00000338761	1456	1300.81	0	0
ENST00000417509	1444	1288.81	7.58582e-08	1.27717e-06
ENST00000484946	610	455.039	2.87905	17.1142
ENST00000490656	660	504.969	1.46703	9.67744
ENST00000439537	1161	1005.81	1.47611	19.3952
ENST00000493251	641	485.994	0.597774	3.79512
ENST00000460127	408	253.708	0	0

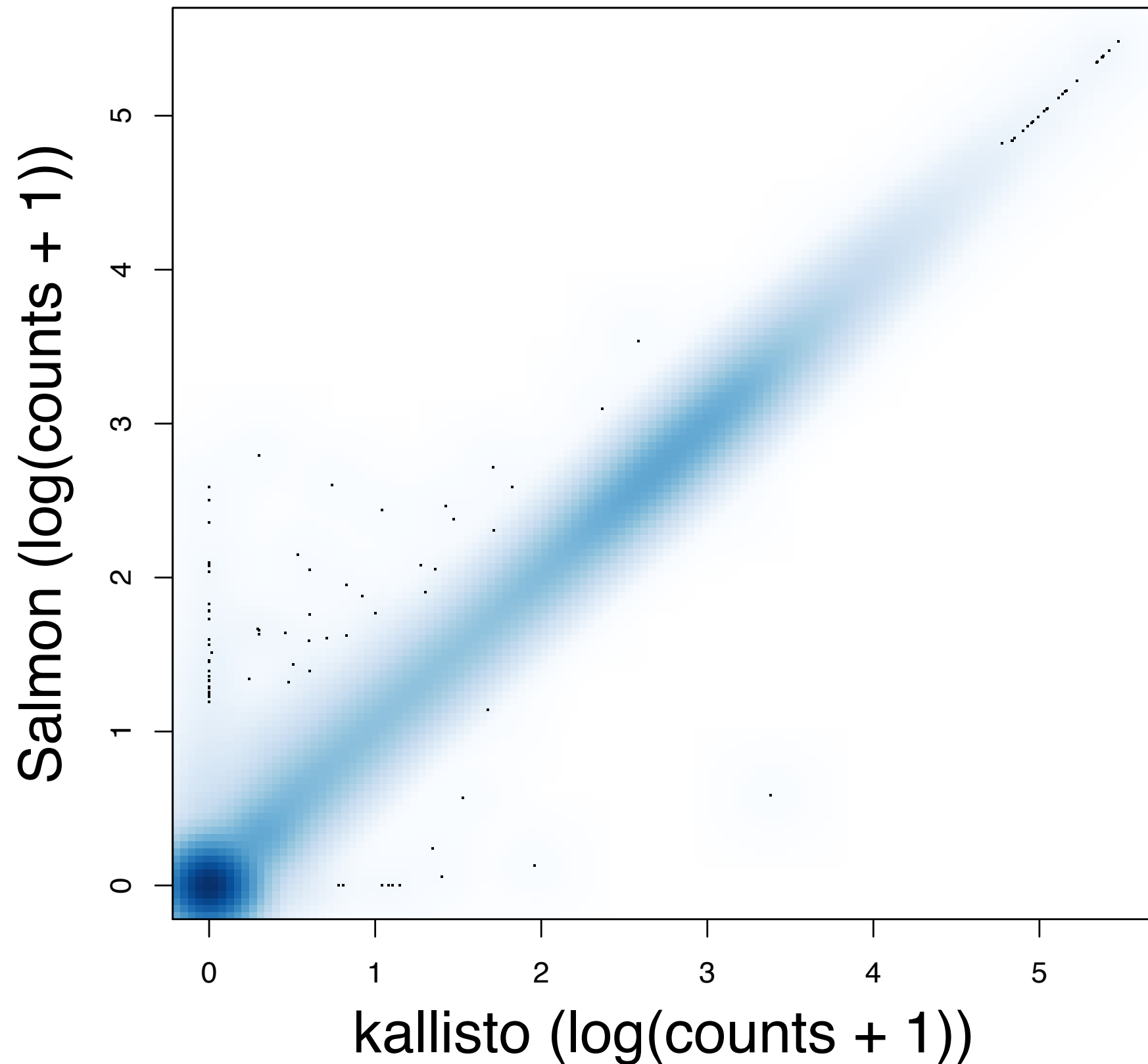
Comparison to traditional workflow

Salmon/kallisto...

- ... are considerably faster than traditional alignment + counting -> allow bootstrapping
- ... provide more highly resolved estimates (transcripts rather than gene) - can be aggregated to gene level
- ... can use a larger fraction of the reads
- ... don't give precise alignments (for e.g. visualization in genome browser) - but avoid large alignment files

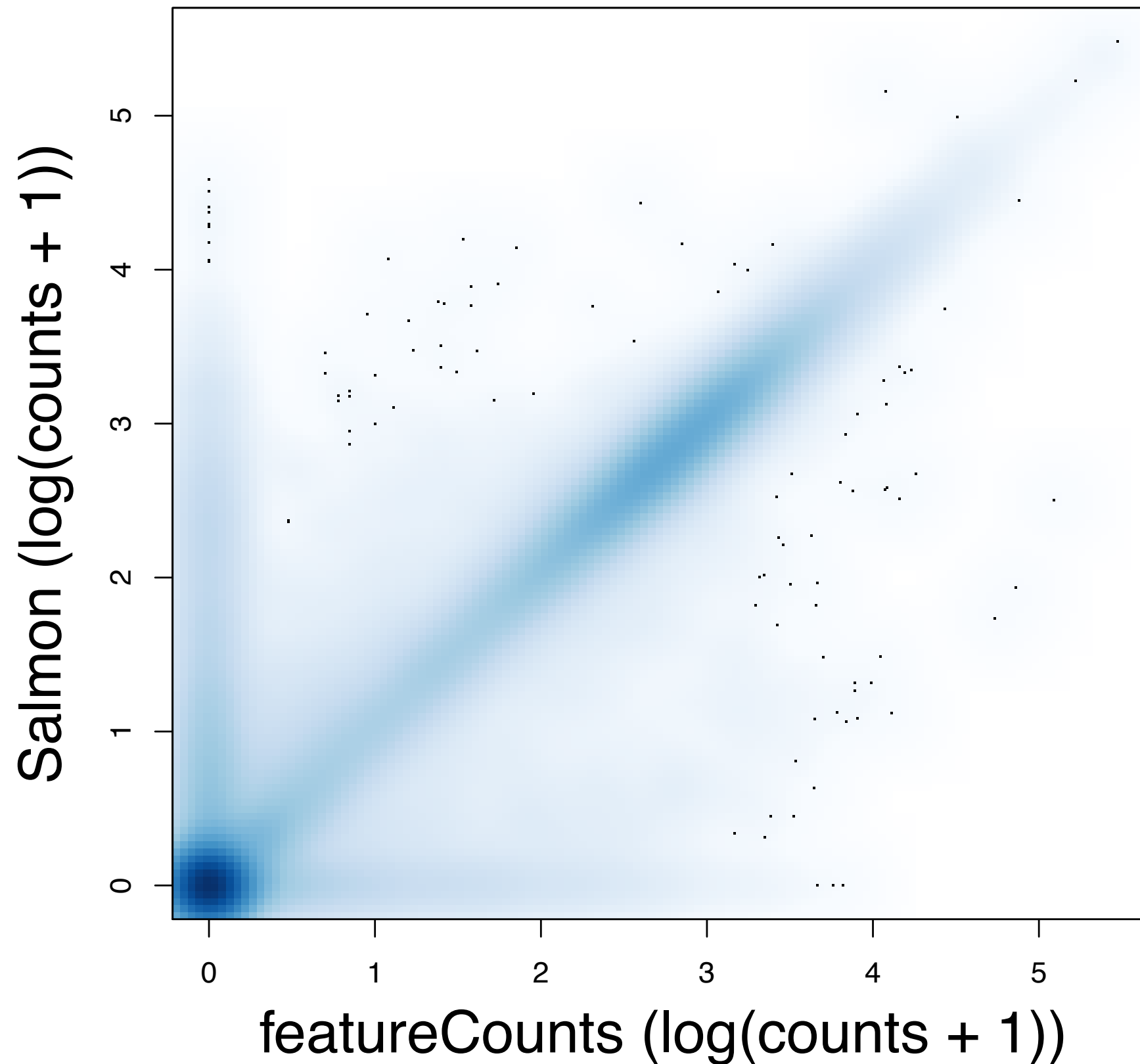
kallisto and Salmon gene counts overall similar

SRR1039508

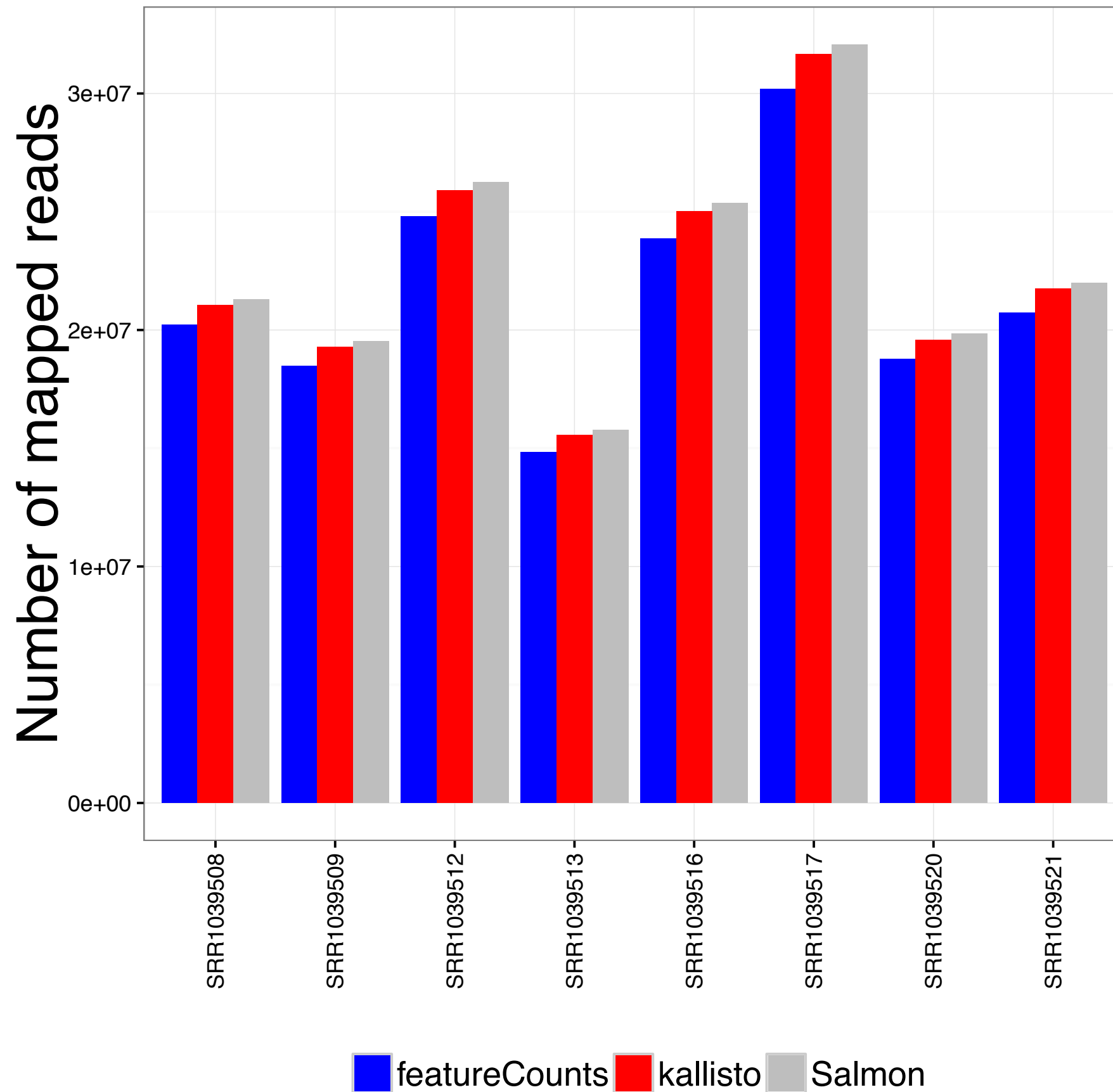


Gene-level counts mostly similar to traditional approach

SRR1039508



kallisto and Salmon can use slightly more reads



How to get the estimated values into R?

```
> library(tximport)
```

```
> salmon_files
```

SRR1039508	SRR1039509
"salmon/SRR1039508/quant.sf"	"salmon/SRR1039509/quant.sf"
SRR1039512	SRR1039513
"salmon/SRR1039512/quant.sf"	"salmon/SRR1039513/quant.sf"
SRR1039516	SRR1039517
"salmon/SRR1039516/quant.sf"	"salmon/SRR1039517/quant.sf"
SRR1039520	SRR1039521
"salmon/SRR1039520/quant.sf"	"salmon/SRR1039521/quant.sf"

```
> head(tx2gene)
```

	tx	gene
1	ENST00000415118	ENSG00000223997
2	ENST00000434970	ENSG00000237235
3	ENST00000448914	ENSG00000228985
4	ENST00000604642	ENSG00000270961
5	ENST00000603326	ENSG00000271317
6	ENST00000604950	ENSG00000270783

How to get the estimated values into R?

```
> txi <- tximport(files = salmon_files, type = "salmon", tx2gene = tx2gene)
reading in files
1 2 3 4 5 6 7 8
summarizing abundance
summarizing counts
summarizing length
> names(txi)
[1] "abundance"          "counts"              "length"
[4] "countsFromAbundance"
```


How to get the estimated values into R?

```
> head(txi$abundance, n = 3)
```

	SRR1039508	SRR1039509	SRR1039512	SRR1039513	SRR1039516
ENSG000000000003	26.95182	19.62924	28.33082	23.24692	36.71688
ENSG000000000005	0.00000	0.00000	0.00000	0.00000	0.00000
ENSG000000000419	38.51888	46.10853	42.34674	43.38094	40.21257

TPMs

	SRR1039517	SRR1039520	SRR1039521
ENSG000000000003	29.09426	34.83193	24.20944
ENSG000000000005	0.00000	0.00000	0.00000
ENSG000000000419	45.72329	39.29645	44.80912

```
> head(txi$counts, n = 3)
```

	SRR1039508	SRR1039509	SRR1039512	SRR1039513	SRR1039516
ENSG000000000003	698.4915	463.0251	895.6865	420.4502	1154.6804
ENSG000000000005	0.0000	0.0000	0.0000	0.0000	0.0000
ENSG000000000419	465.9998	515.5963	625.0002	365.6836	590.0994

counts

	SRR1039517	SRR1039520	SRR1039521
ENSG000000000003	1078.464	780.3976	589.2203
ENSG000000000005	0.000	0.0000	0.0000
ENSG000000000419	797.987	419.6755	510.9196

```
> head(txi$length, n = 3)
```

	SRR1039508	SRR1039509	SRR1039512	SRR1039513	SRR1039516
ENSG000000000003	1983.8737	1947.5904	1978.7880	1993.6675	1963.7941
ENSG000000000005	783.3978	783.3978	783.3978	783.3978	783.3978
ENSG000000000419	926.0907	923.2618	923.7694	929.2005	916.3488

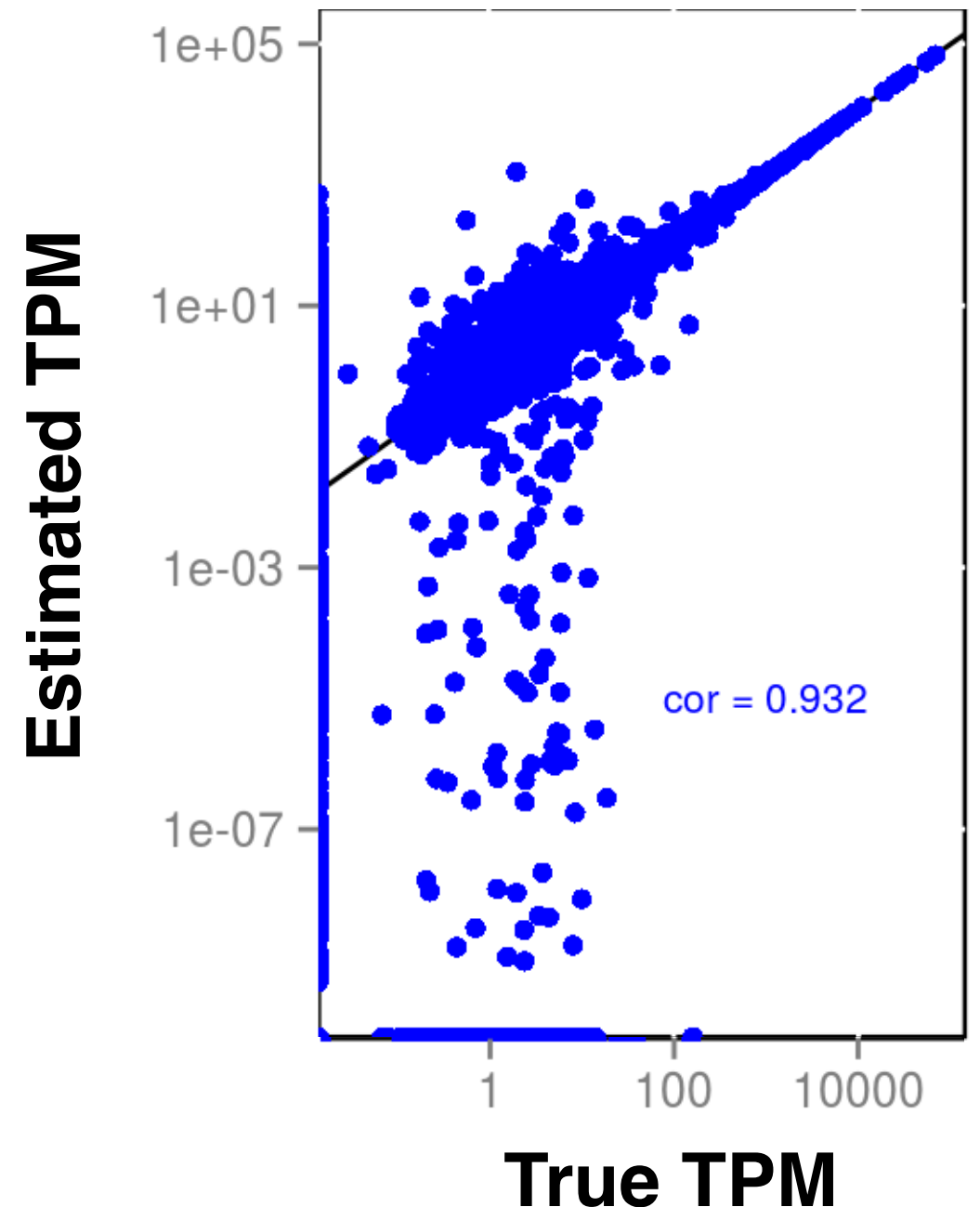
**“ATL”
offsets**

	SRR1039517	SRR1039520	SRR1039521
ENSG000000000003	1967.1231	1951.0682	1986.9260
ENSG000000000005	783.3978	783.3978	783.3978
ENSG000000000419	926.1689	930.0241	930.8409



A word of warning

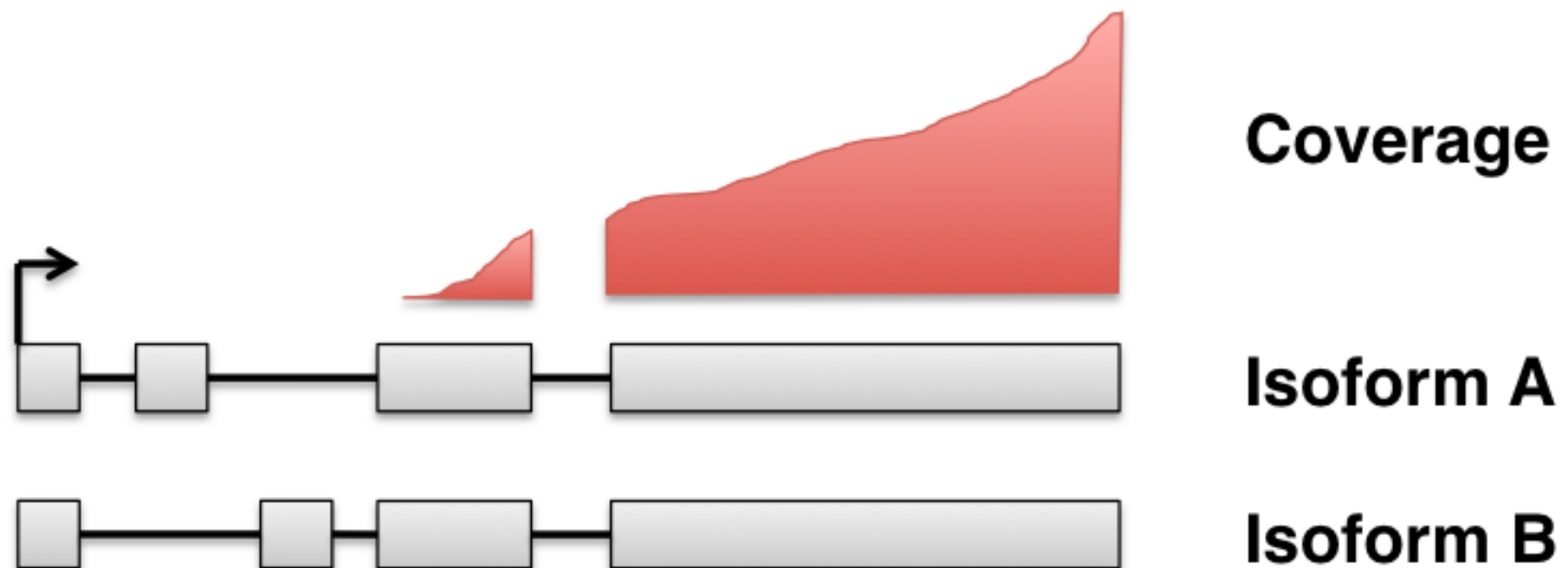
- Abundance estimates for lowly expressed transcripts are highly variable and should be interpreted with caution





A word of warning

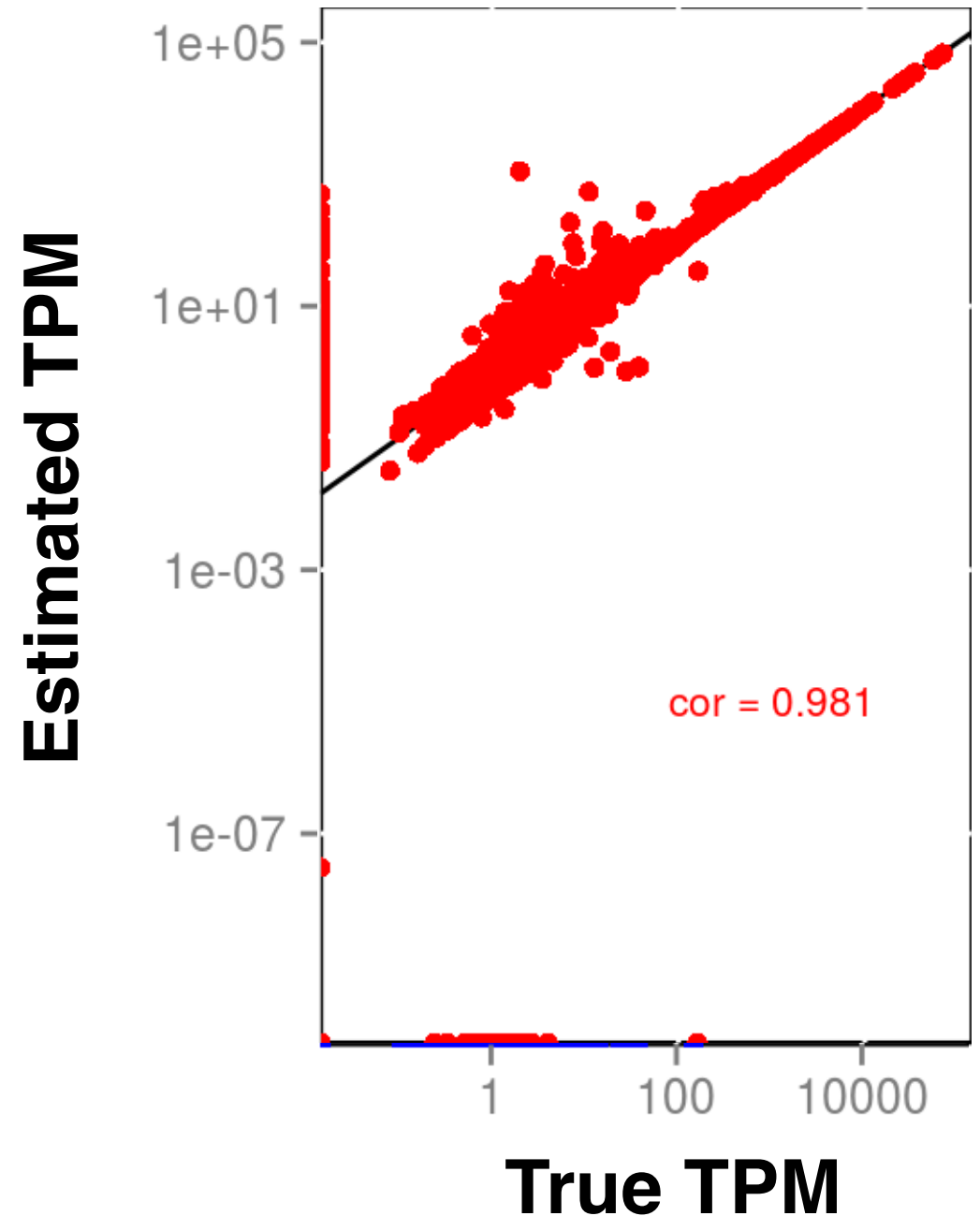
- Problematic when coverage of region defining an isoform is low





A word of warning

- When aggregated to the gene level, abundance estimates are less variable



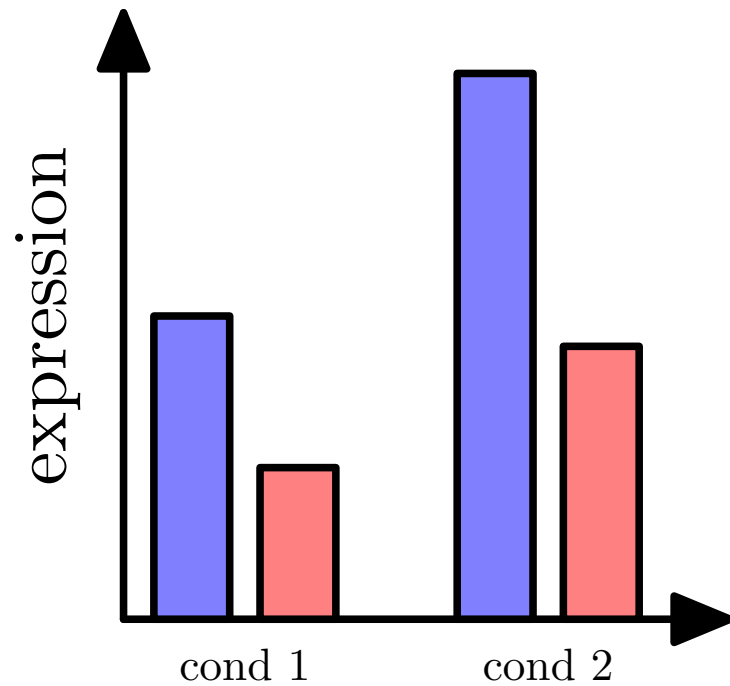
Differential analysis types for RNA-seq

- Has the total output of a gene changed? **DGE**
- Has the expression of individual transcripts changed? **DTE**
- Has *any* isoform of a given gene changed? **DTE+G**
- Has the isoform composition for a given gene changed? **DTU/DEU**

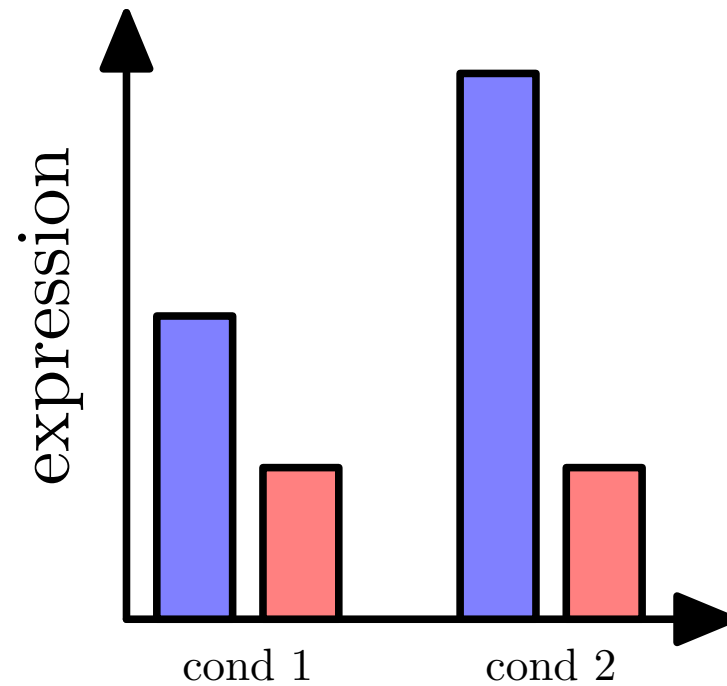
- need **different** abundance quantification of transcriptomic features (genes, transcripts, exons)

Differential analysis types for RNA-seq

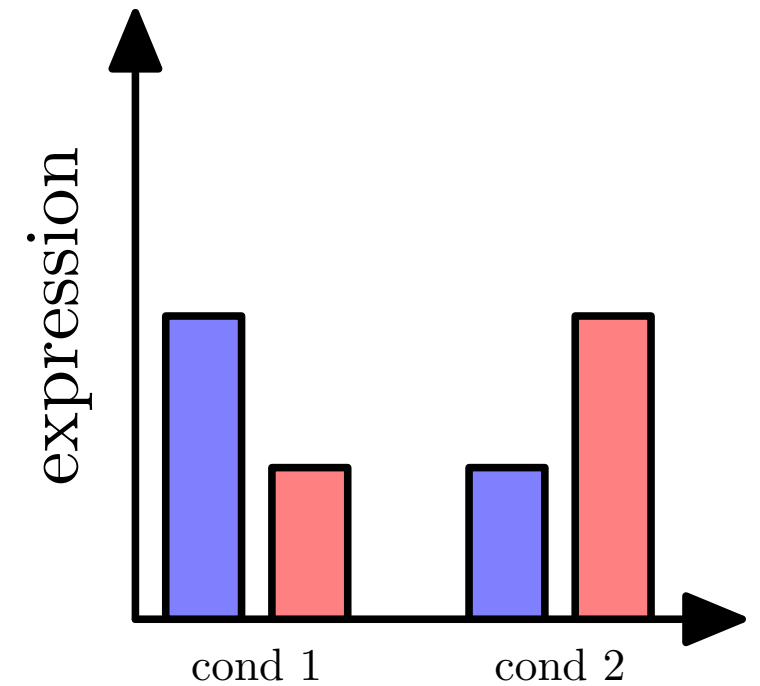
DGE
(also DTE)



DTE
(also DGE, DTU)



DTU
(also DTE)



 isoform A  isoform B

References

- Srivastava et al.: RapMap: a rapid, sensitive and accurate tool for mapping RNA-seq read to transcriptomes. Bioinformatics 32:i192-i200 (2016) - **RapMap**
- Patro et al.: Accurate, fast, and model-aware transcript expression quantification with Salmon. bioRxiv <http://dx.doi.org/10.1101/021592> (2015) - **Salmon**
- Bray et al.: Near-optimal probabilistic RNA-seq quantification. Nature Biotechnology 34(5):525-527 (2016) - **kallisto**
- Patro et al.: Sailfish enables alignment-free isoform quantification from RNA-seq reads using lightweight algorithms. Nature Biotechnology 32:462-464 (2014) - **Sailfish**
- Pimentel et al.: Differential analysis of RNA-Seq incorporating quantification uncertainty. bioRxiv <http://dx.doi.org/10.1101/058164> (2016) - **sleuth**
- Wagner et al.: Measurement of mRNA abundance using RNA-seq data: RPKM measure is inconsistent among samples. Theory in Biosciences 131:281-285 (2012) - **TPM vs FPKM**
- Soneson et al.: Differential analyses for RNA-seq: transcript-level estimates improve gene-level inferences. F1000Research 4:1521 (2016) - **ATL offsets (tximport package)**
- Li et al.: RNA-seq gene expression estimation with read mapping uncertainty. Bioinformatics 26(4):493-500 (2010) - **TPM, RSEM**