Transcriptomic Data Analysis

Jędrzej Jakub Szymański

IPK Gatersleben & FZJ Jülich

Transcriptomic Data Analysis

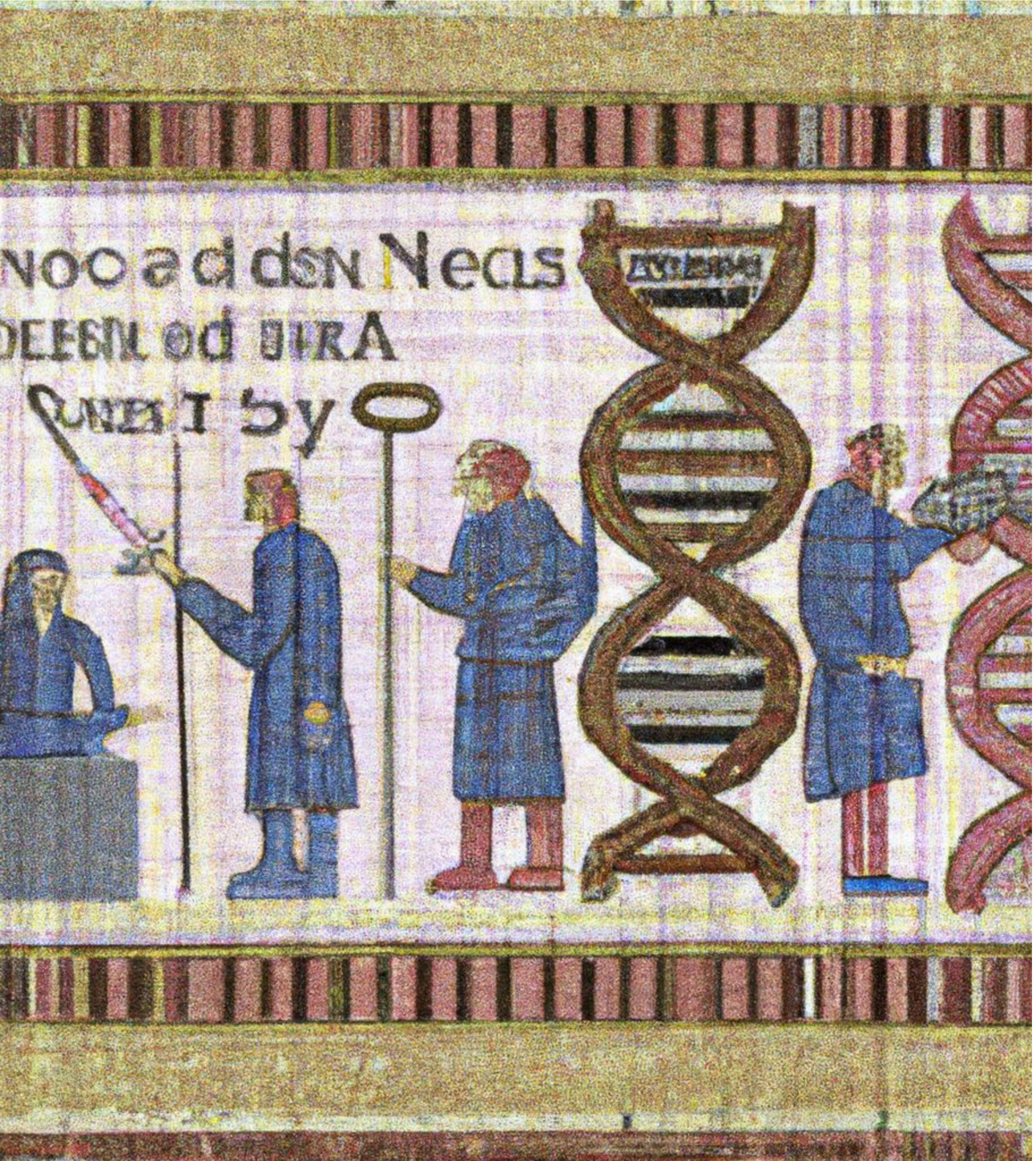
Jędrzej Jakub Szymański

IPK Gatersleben & FZJ Jülich

“Transcriptomic data analysis in the middle ages” DALL-E2 generative AI model

Lectures in the series

1. Introduction to Transcriptomic Data
2. Study Design, Annotation, and Data Import
3. Filtering and Normalization of Transcriptomic Data
4. Data Analysis - Multivariate Statistics and Visualization
5. Accessing Public Data
6. Differential Gene Expression
7. Functional Enrichment Analysis



Introduction to Transcriptomic Data

What we **can** learn from transcriptomic data?

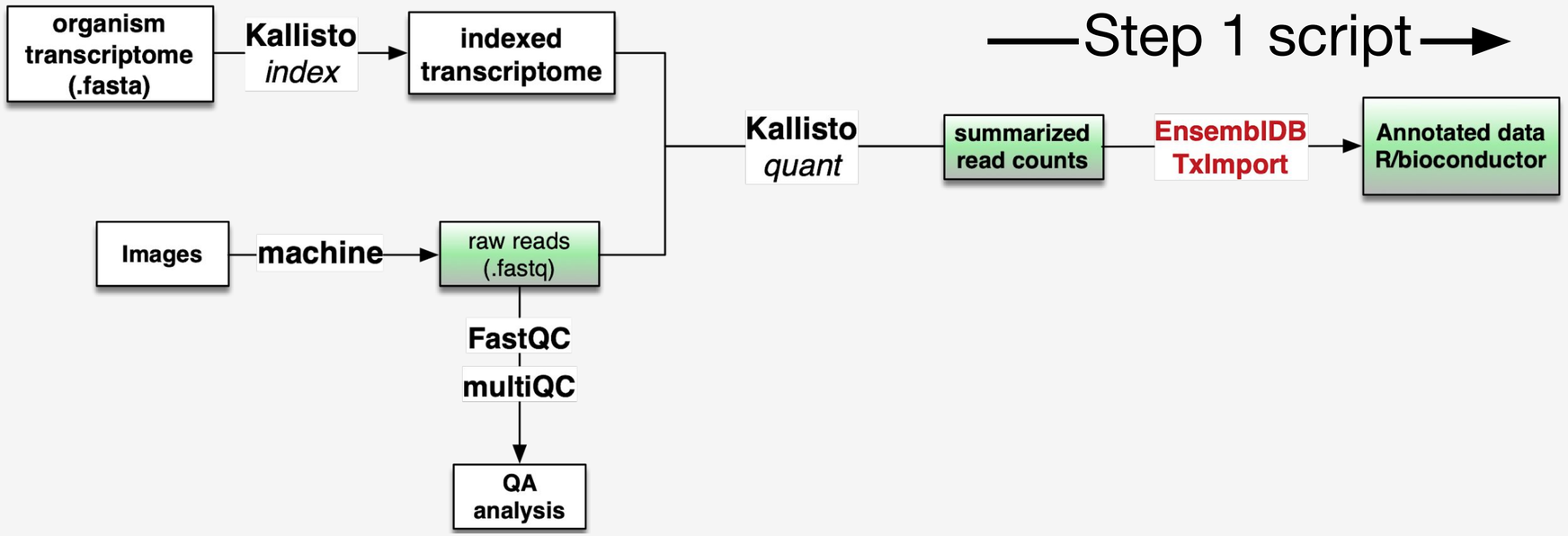
Introduction to data formats

Basic definitions - read, read count, transcript length, RPKM, TPM

Start an RStudio Project directory that we'll use for the rest of the course.

Open and discuss our first script, including installation of packages

Tracking our workflow in this course



Don't worry if you've struggled with QA or read mapping!

Explore other Kallisto functions on your own

```
(base) daniels-mbp-3:course_dataset danielbeiting$ kallisto
kallisto 0.46.2

Usage: kallisto <CMD> [arguments] ..

Where <CMD> can be one of:

    index         Builds a kallisto index
    quant         Runs the quantification algorithm
    bus           Generate BUS files for single-cell data
    pseudo        Runs the pseudoalignment step
    merge         Merges several batch runs
    h5dump        Converts HDF5-formatted results to plaintext
    inspect       Inspects and gives information about an index
    version       Prints version information
    cite          Prints citation information
```

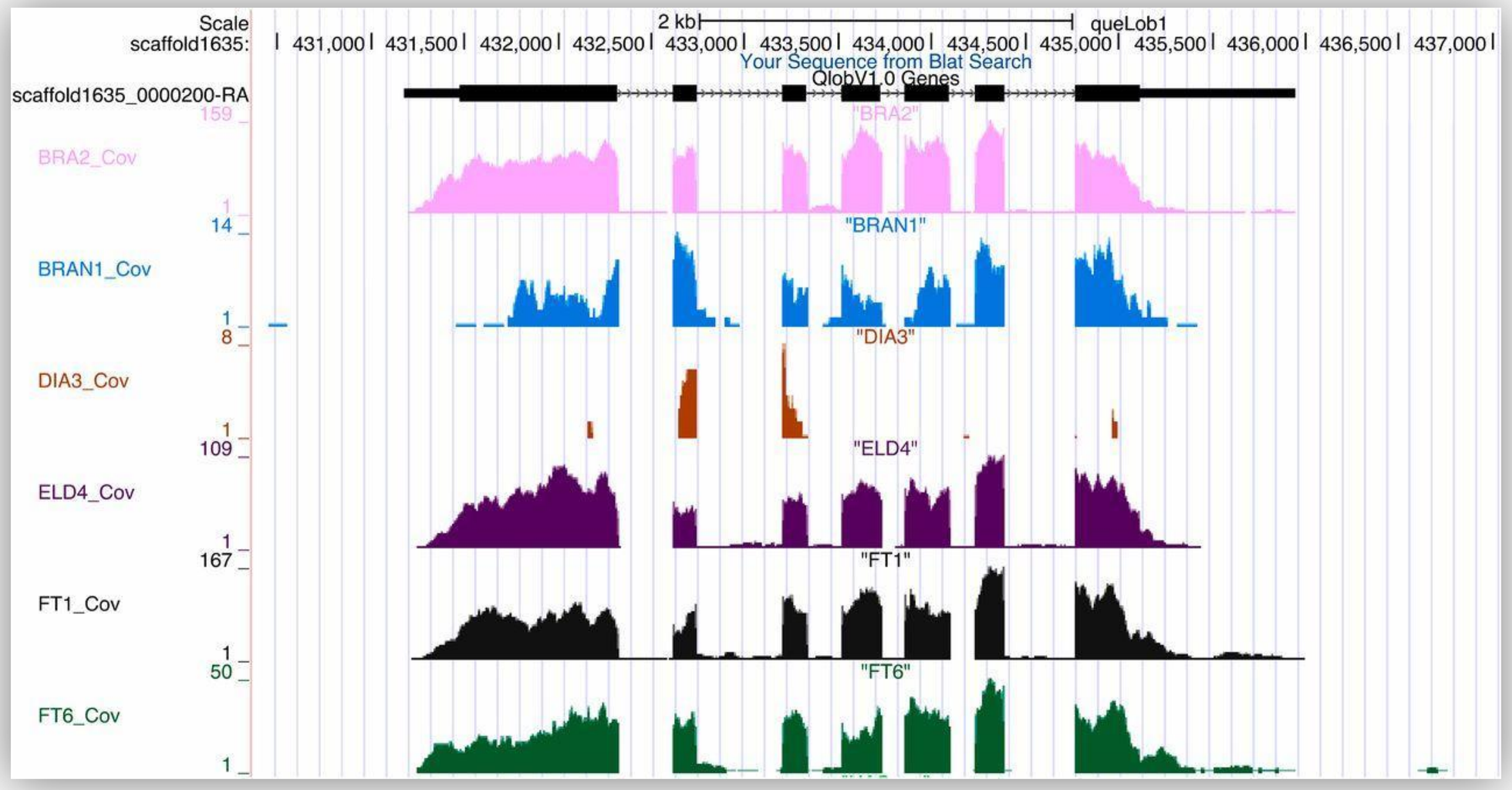
```
kallisto 0.46.0
Computes equivalence classes for reads and quantifies abundances

Usage: kallisto quant [arguments] FASTQ-files

Required arguments:
-i, --index=STRING            Filename for the kallisto index to be used for
                              quantification
-o, --output-dir=STRING       Directory to write output to

Optional arguments:
    --bias                    Perform sequence based bias correction
-b, --bootstrap-samples=INT   Number of bootstrap samples (default: 0)
    --seed=INT                Seed for the bootstrap sampling (default: 42)
    --plaintext               Output plaintext instead of HDF5
    --fusion                  Search for fusions for Pizzly
    --single                  Quantify single-end reads
    --single-overhang         Include reads where unobserved rest of fragment is
                              predicted to lie outside a transcript
    --fr-stranded             Strand specific reads, first read forward
    --rf-stranded             Strand specific reads, first read reverse
-l, --fragment-length=DOUBLE  Estimated average fragment length
-s, --sd=DOUBLE               Estimated standard deviation of fragment length
                              (default: -l, -s values are estimated from paired
                               end data, but are required when using --single)
-t, --threads=INT             Number of threads to use (default: 1)
    --pseudobam               Save pseudoalignments to transcriptome to BAM file
    --genomebam               Project pseudoalignments to genome sorted BAM file
-g, --gtf                     GTF file for transcriptome information
                              (required for --genomebam)
-c, --chromosomes             Tab separated file with chromosome names and lengths
                              (optional for --genomebam, but recommended)
```

Using Kallisto (pseudoalignment) is a flexible strategy for handling many types of seq experiments



Pseudoalignment also performs well for lowly expressed transcripts (e.g. lncRNAs)

Benchmark of long non-coding RNA quantification for RNA sequencing of cancer samples

Hong Zheng [1], Kevin Brennan [1], Mikel Hernaez [2] and Olivier Gevaert [1,3,*]

[1]Stanford Center for Biomedical Informatics Research, Department of Medicine, Stanford University, 1265
University of Illinois at
Biomedical Data Science,

"In this benchmarking study, we compared the performance of pseudoalignment methods Kallisto and Salmon, and alignment based methods HTSeq, featureCounts, and RSEM, in lncRNA quantification, by applying them to a simulated RNA-Seq dataset and a pan-cancer RNA-Seq dataset from TCGA."

"In summary, pseudoalignment methods Kallisto or Salmon in combination with the full transcriptome annotation is our recommended strategy for RNA-Seq analysis for lncRNAs."

Zheng et al, GigaScience, 2019

protocols.hostmicrobe.org/compute/kallisto

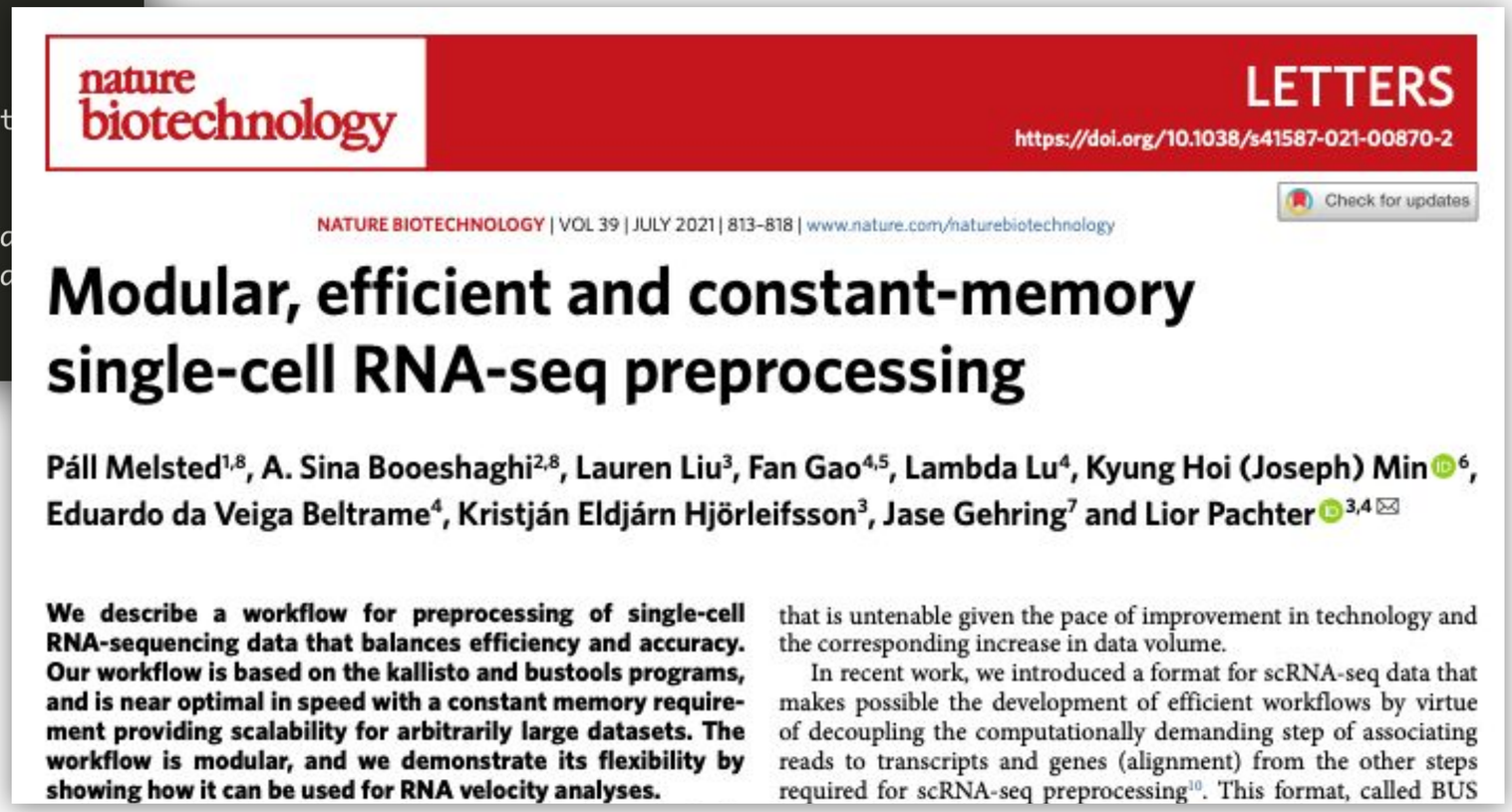
Kallisto ‘bus tools’ for scRNAseq

```
(base) daniels-mbp-3:course_dataset danielbeiting$ kallisto
kallisto 0.46.2

Usage: kallisto <CMD> [arguments] ..

Where <CMD> can be one of:

    index         Builds a kallisto index
    quant         Runs the quantification algorithm
    bus           Generate BUS files for single-cell dat
    pseudo        Runs the pseudoalignment step
    merge         Merges several batch runs
    h5dump        Converts HDF5-formatted results to pla
    inspect       Inspects and gives information about a
    version       Prints version information
    cite          Prints citation information
```



nature biotechnology

LETTERS

https://doi.org/10.1038/s41587-021-00870-2

Check for updates

NATURE BIOTECHNOLOGY | VOL 39 | JULY 2021 | 813–818 | www.nature.com/naturebiotechnology

Modular, efficient and constant-memory single-cell RNA-seq preprocessing

Páll Melsted[1,8], A. Sina Booeshaghi[2,8], Lauren Liu[3], Fan Gao[4,5], Lambda Lu[4], Kyung Hoi (Joseph) Min[6], Eduardo da Veiga Beltrame[4], Kristján Eldjárn Hjörleifsson[3], Jase Gehring[7] and Lior Pachter[3,4] ✉

We describe a workflow for preprocessing of single-cell RNA-sequencing data that balances efficiency and accuracy. Our workflow is based on the kallisto and bustools programs, and is near optimal in speed with a constant memory requirement providing scalability for arbitrarily large datasets. The workflow is modular, and we demonstrate its flexibility by showing how it can be used for RNA velocity analyses.

that is untenable given the pace of improvement in technology and the corresponding increase in data volume.

In recent work, we introduced a format for scRNA-seq data that makes possible the development of efficient workflows by virtue of decoupling the computationally demanding step of associating reads to transcripts and genes (alignment) from the other steps required for scRNA-seq preprocessing[10]. This format, called BUS

Kallisto ‘bus tools’ for scRNAseq

```
(base) daniels-mbp-3:course_dataset danielbeiting$ kallisto
kallisto 0.46.2

Usage: kallisto <CMD> [arguments] ..

Where <CMD> can be one of:

    index         Builds a kallisto index
    quant         Runs the quantification algorithm
    bus           Generate BUS files for single-cell
    pseudo        Runs the pseudoalignment step
    merge         Merges several batch runs
    h5dump        Converts HDF5-formatted results to
    inspect
    version
    cite
```

“Our scRNA-seq workflow is up to 51 times faster than CellRanger…[and] requires a small fixed amount of constant memory that is independent of the number of reads being pre-processed. It is therefore suitable for low-cost and environmentally conscious cloud computing”

Cold Spring Harbor Laboratory

bioRχiv

THE PREPRINT SERVER FOR BIOLOGY

Modular and efficient pre-processing of single-cell RNA-seq

Posted July 26, 2019.

Download PDF
Supplementary Material
Data/Code
XML
Revision Summary

Email
Share
Citation Tools

PDF

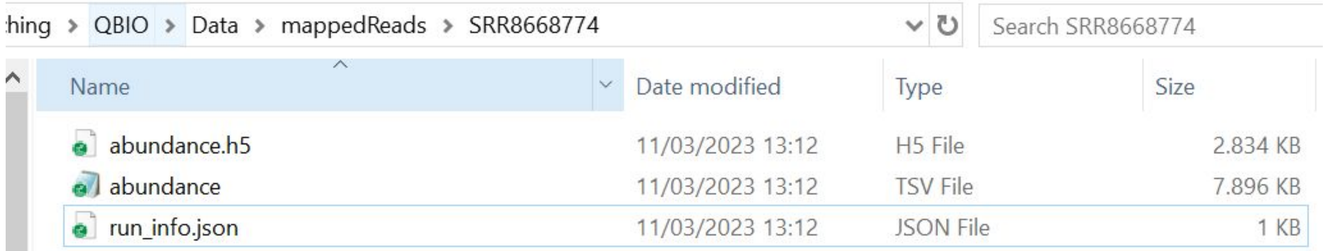
Analysis of single-cell RNA-seq data begins with pre-processing of sequencing reads to generate count matrices. We investigate algorithm choices for the challenges of pre-

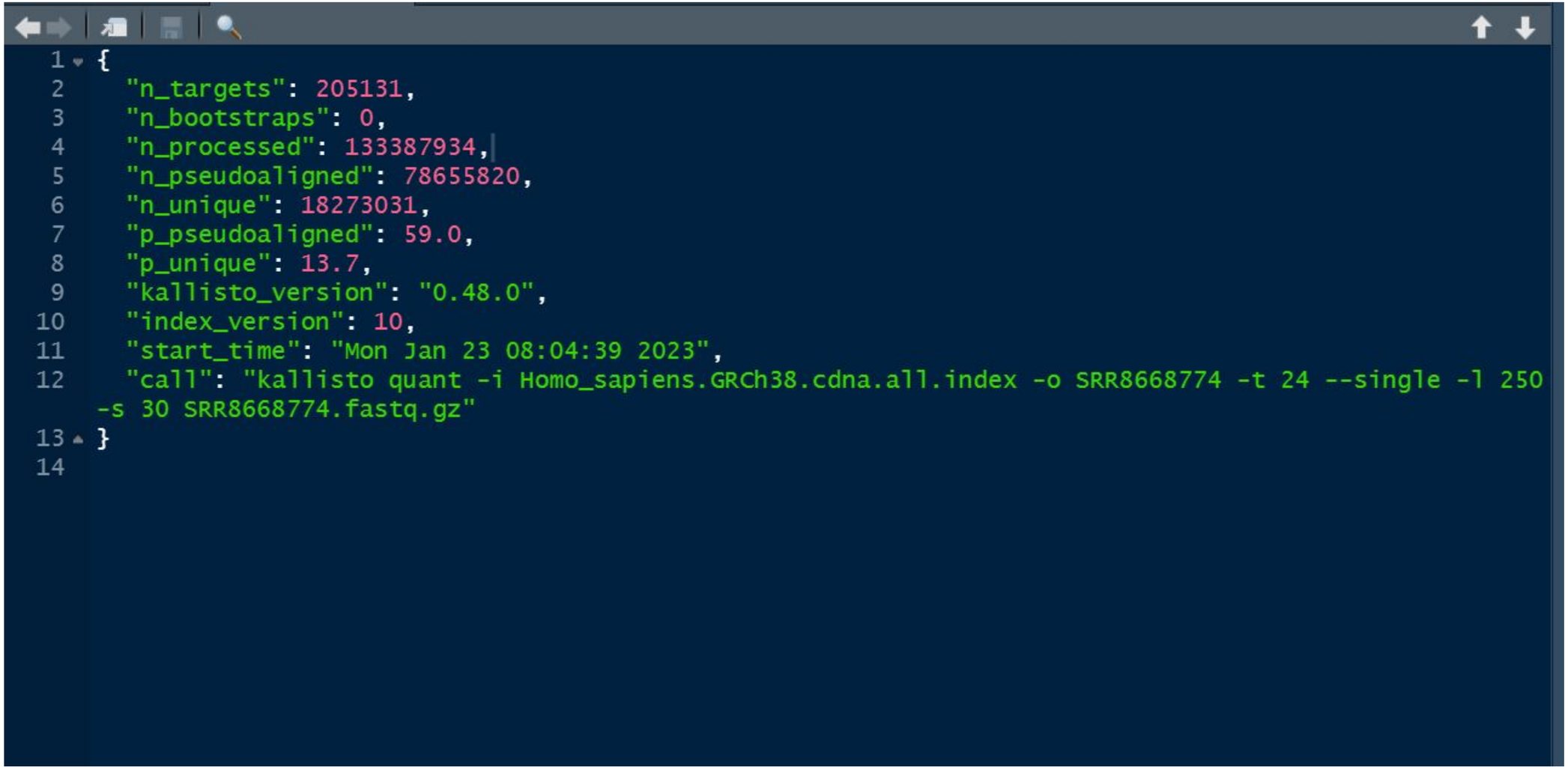
Processed ~400M scRNAseq reads in ~16 min, compared to ~16 hours with Cell Ranger.

is modular, and we demonstrate its flexibility by showing how it can be used for RNA velocity analyses. Documentation and tutorials for using the kallisto | bus workflow are available at https://www.kallistobus.tools/.

Open Kallisto output

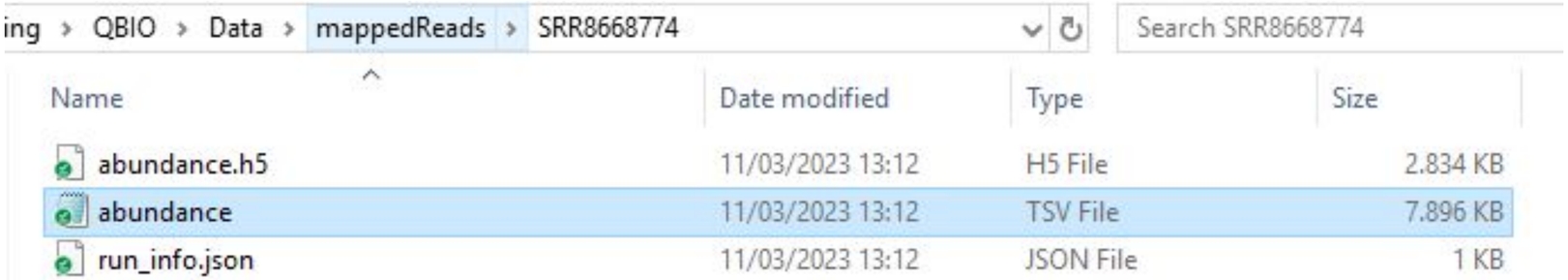
run_info.json


hing
QBIO
Data
mappedReads
SRR8668774
Search SRR8668774
Name
Date modified
Type
Size
abundance.h5
11/03/2023 13:12
H5 File
2.834 KB
abundance
11/03/2023 13:12
TSV File
7.896 KB
run_info.json
11/03/2023 13:12
JSON File
1 KB


{
"n_targets": 205131,
"n_bootstraps": 0,
"n_processed": 133387934,
"n_pseudoaligned": 78655820,
"n_unique": 18273031,
"p_pseudoaligned": 59.0,
"p_unique": 13.7,
"kallisto_version": "0.48.0",
"index_version": 10,
"start_time": "Mon Jan 23 08:04:39 2023",
"call": "kallisto quant -i Homo_sapiens.GRCh38.cdna.all.index -o SRR8668774 -t 24 --single -l 250 -s 30 SRR8668774.fastq.gz"
}

Open Kallisto output

abundance.tsv



abundance - Notepad

File Edit Format View Help

target_id	length	eff_length	est_counts	tpm
ENST00000631435.1	12	3.83025	0	0
ENST00000415118.1	8	3.3445	0	0
ENST00000448914.1	13	3.91173	0	0
ENST00000434970.2	9	3.4951	0	0
ENST00000632684.1	12	3.83025	0	0
ENST00000633010.1	16	4.09465	0	0
ENST00000633009.1	20	4.24848	0	0
ENST00000632524.1	11	3.73525	0	0
ENST00000633353.1	31	4.49022	0	0
ENST00000633765.1	31	4.49022	0	0
ENST00000633159.1	21	4.27752	0	0
ENST00000631884.1	17	4.14041	0	0
ENST00000634070.1	18	4.18068	0	0
ENST00000633504.1	31	4.49022	0	0
ENST00000632542.1	18	4.18068	0	0
ENST00000632619.1	28	4.43409	0	0
ENST00000631895.1	23	4.32879	0	0
ENST00000633030.1	19	4.21642	0	0
ENST00000632911.1	31	4.49022	0	0
ENST00000632968.1	17	4.14041	0	0
ENST00000632963.1	20	4.24848	0	0
ENST00000632473.1	31	4.49022	0	0
ENST00000633968.1	20	4.24848	0	0
ENST00000634085.1	16	4.09465	0	0

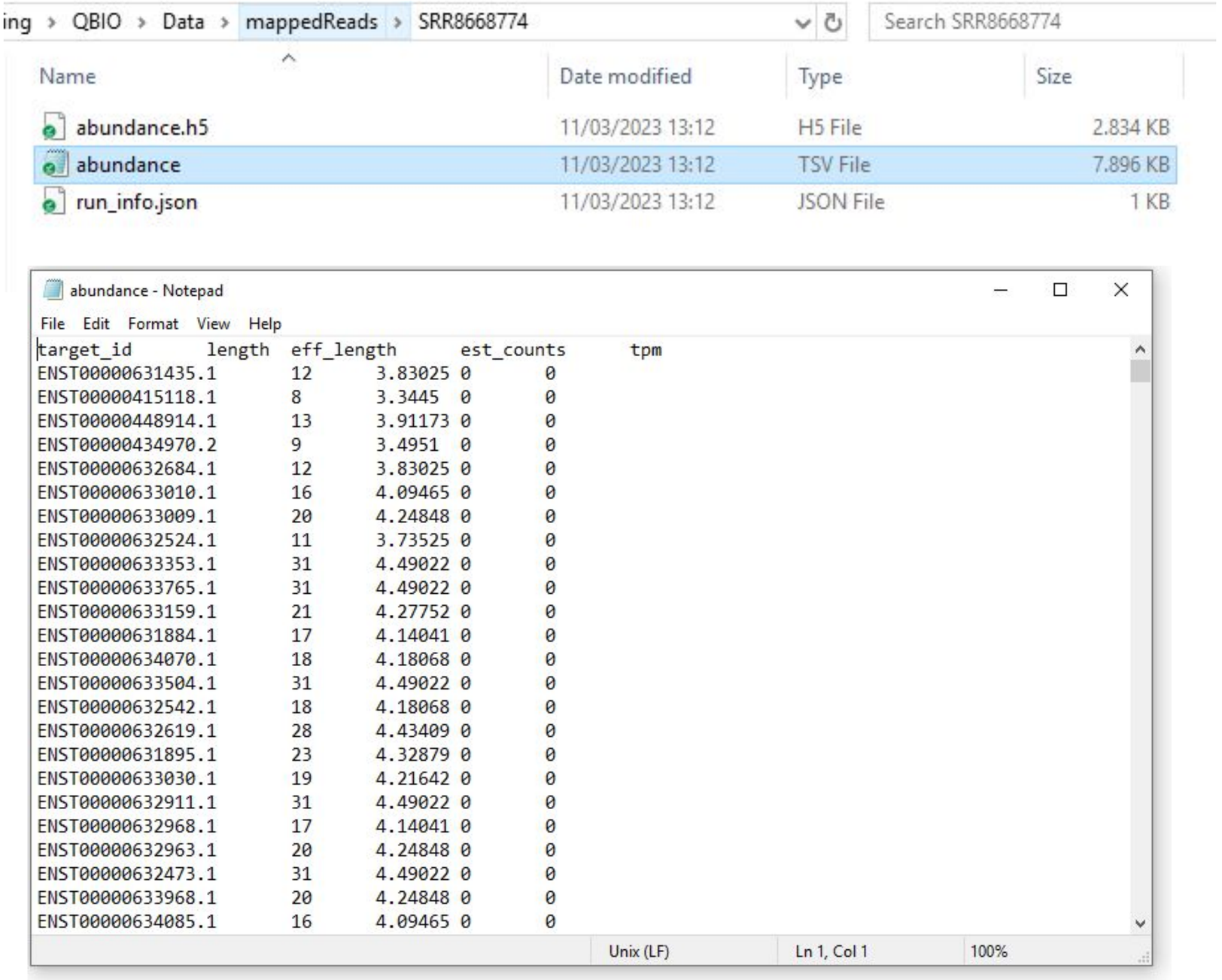
Unix (LF)

abundance - Notepad

File Edit Format View Help

ENST00000371754.8	5235	4986	1208.55	4.16607
ENST00000469991.1	640	391	14.5506	0.639615
ENST00000371752.5	7212	6963	6866.66	16.9498
ENST00000396105.6	7209	6960	12876.7	31.7988
ENST00000371744.5	2964	2715	1972.76	12.4887
ENST00000455070.1	2122	1873	34.6566	0.318026
ENST00000509231.1	2224	1975	0	0
ENST00000646235.1	3341	3092	0	0
ENST00000644105.2	2821	2572	0	0
ENST00000646118.1	3411	3162	3	0.016307
ENST00000647304.1	3707	3458	1075.39	5.3451
ENST00000395409.7	2192	1943	0	0
ENST00000540171.2	1307	1058	2	0.0324907
ENST00000675743.1	648	399	0	0
ENST00000674723.1	931	682	0	0
ENST00000674869.1	837	588	0	0
ENST00000675248.1	577	328	0	0
ENST00000676427.1	883	634	32.2398	0.874013
ENST00000330634.11	7578	7329	289.566	0.679076
ENST00000392634.9	7623	7374	0.956435	0.00222929
ENST00000675482.1	1020	771	20.6712	0.460816
ENST00000398337.8	1689	1440	224.596	2.68074
ENST00000674966.1	859	610	8.63282	0.243242
ENST00000675207.1	7732	7483	58.9235	0.13534
ENST00000675616.1	431	182	0	0

Unix (LF) Ln 1, Col 1 100%


ing › QBIO › Data › mappedReads › SRR8668774
Search SRR8668774
Name
Date modified
Type
Size
abundance.h5
11/03/2023 13:12
H5 File
2.834 KB
abundance
11/03/2023 13:12
TSV File
7.896 KB
run_info.json
11/03/2023 13:12
JSON File
1 KB
abundance - Notepad
File Edit Format View Help
target_id length eff_length est_counts tpm
ENST00000631435.1 12 3.83025 0 0
ENST00000415118.1 8 3.3445 0 0
ENST00000448914.1 13 3.91173 0 0
ENST00000434970.2 9 3.4951 0 0
ENST00000632684.1 12 3.83025 0 0
ENST00000633010.1 16 4.09465 0 0
ENST00000633009.1 20 4.24848 0 0
ENST00000632524.1 11 3.73525 0 0
ENST00000633353.1 31 4.49022 0 0
ENST00000633765.1 31 4.49022 0 0
ENST00000633159.1 21 4.27752 0 0
ENST00000631884.1 17 4.14041 0 0
ENST00000634070.1 18 4.18068 0 0
ENST00000633504.1 31 4.49022 0 0
ENST00000632542.1 18 4.18068 0 0
ENST00000632619.1 28 4.43409 0 0
ENST00000631895.1 23 4.32879 0 0
ENST00000633030.1 19 4.21642 0 0
ENST00000632911.1 31 4.49022 0 0
ENST00000632968.1 17 4.14041 0 0
ENST00000632963.1 20 4.24848 0 0
ENST00000632473.1 31 4.49022 0 0
ENST00000633968.1 20 4.24848 0 0
ENST00000634085.1 16 4.09465 0 0
Unix (LF)
Ln 1, Col 1
100%

The ‘effective length’ of a transcript

The length of a transcript after adjusting for the total number of possible positions a fragment of size X could originate from

$$L_{effective} = L_{actual} - L_{fragment} + 1$$

transcript	ATGCGTAACATG	$L_{actual} = 12$	**$L_{effective} = 10$**
fragment	NNN	$L_{fragment} = 3$	

RNAseq gives relative quantification of gene expression

"All commonly used techniques to measure mRNA abundance, including qPCR, microarray signals, as well as reads per kilobase per million reads (RPKM) for RNAseq data, aim at estimating a statistic that is as closely proportional to the **relative molar concentration** as possible."

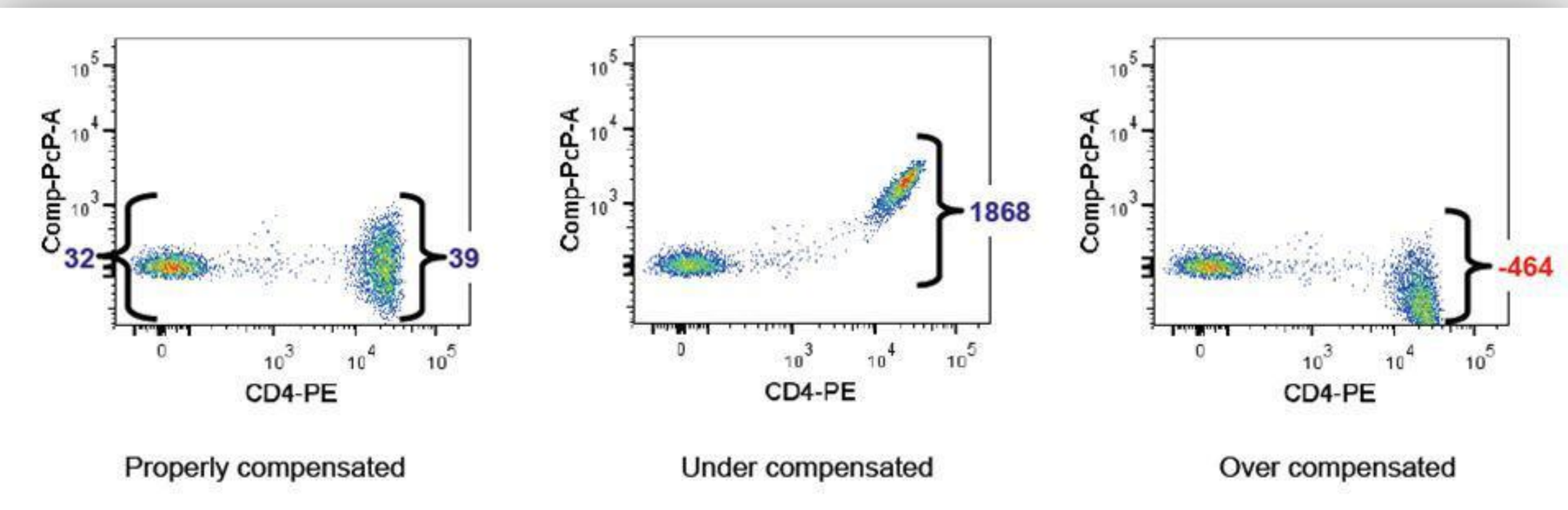
- Wagner, Theories in Biosci., 2012

Understanding units of measurement for RNAseq is critical to understanding how we determine differential expression

Normalization

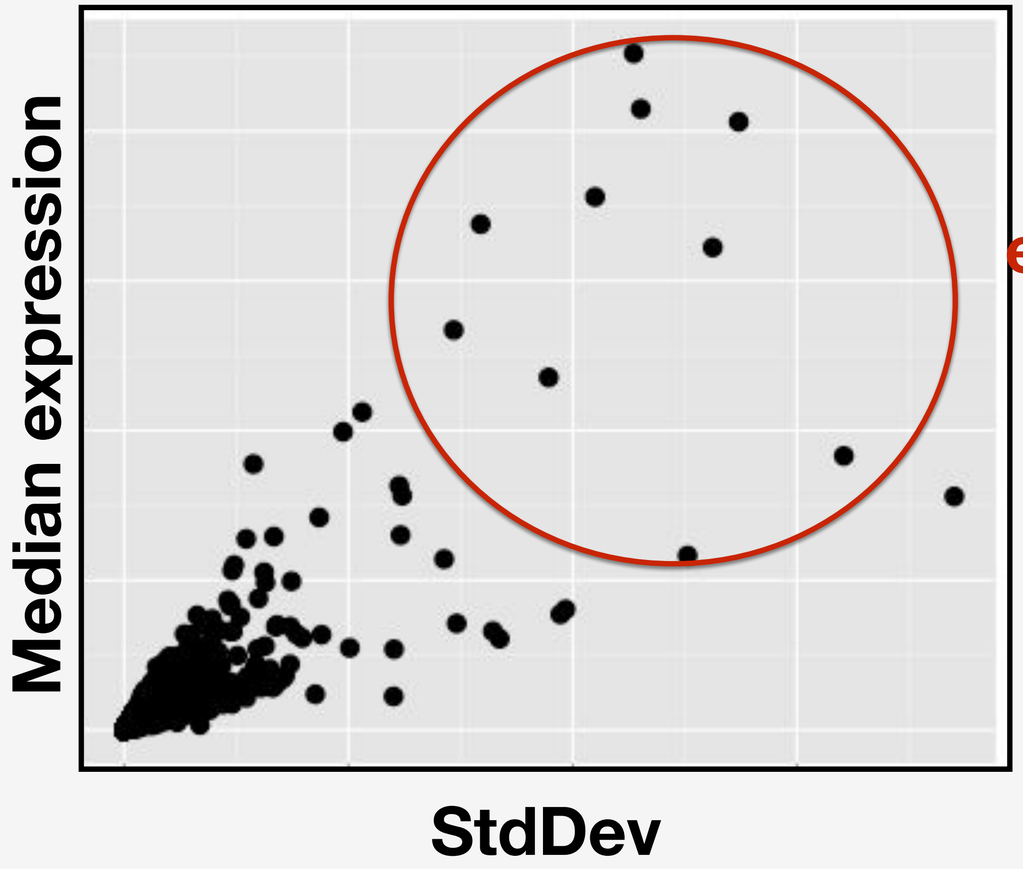
generic term referring to any number of ways that a dataset is globally altered to improve our ability to detect DEGs

RNAseq normalization ≈ Flow cytometry compensation



raw counts

	mouse 1	mouse 2	mouse 3	mouse 4	mouse 5	StDev
gene A	5	10	10	15	10	***3.5***
gene B	115	110	100	115	118	***7.1***
gene C	1000	1100	1050	1045	1030	***36.4***
gene D	8000	9000	10000	6000	7030	***1576.4***



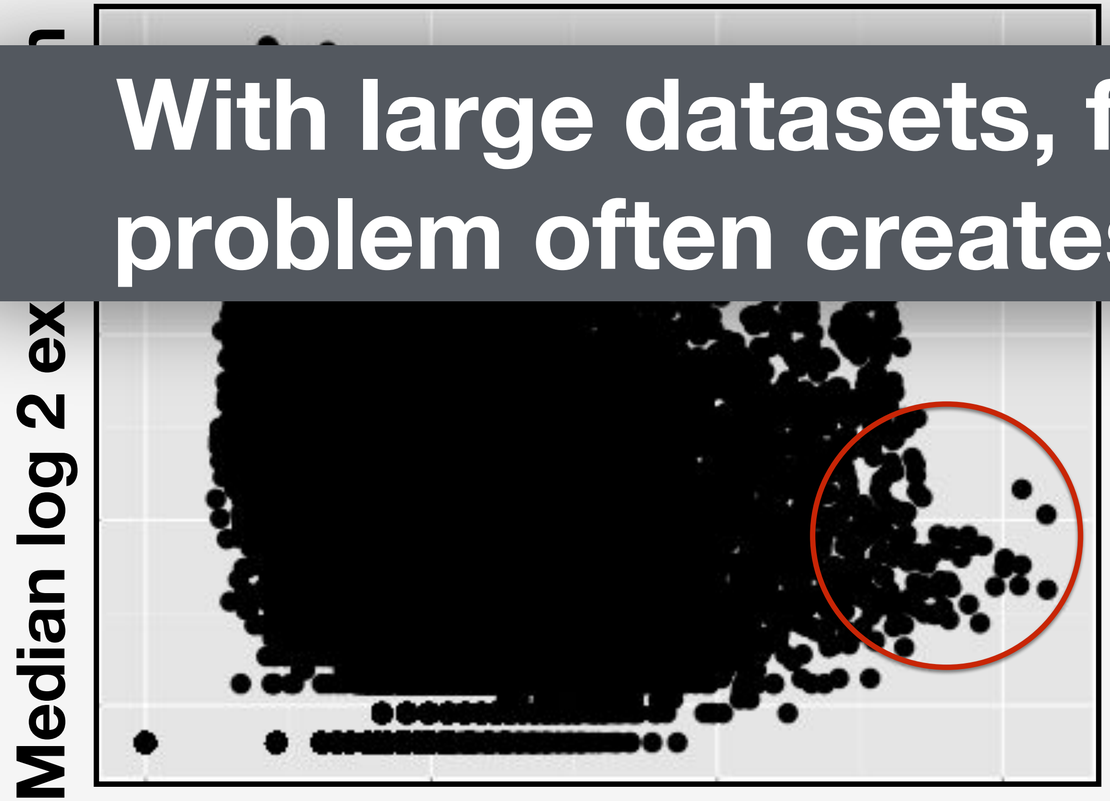
genes with higher expression have higher stdDev

Heteroscedasticity

Log 2

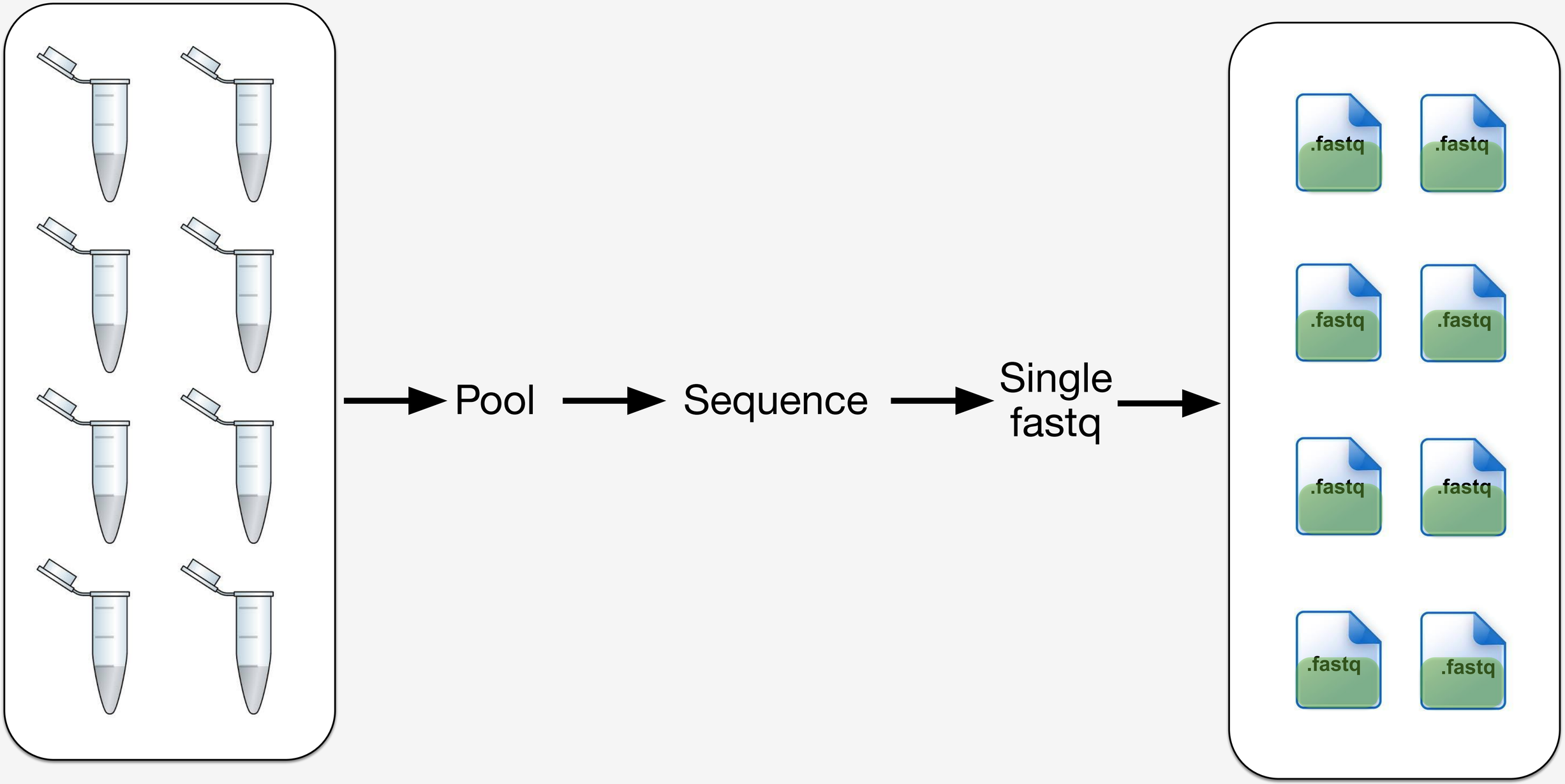
	mouse 1	mouse 2	mouse 3	mouse 4	mouse 5	StDev
gene A	2.3	3.3	3.3	3.9	3.3	*0.57*
gene B	6.8	6.8	6.6	6.8	6.9	*0.09*
gene C	10.0	10.1	10.0	10.0	10.0	*0.05*
gene D	13.0	13.1	13.3	12.6	12.8	*0.29*

With large datasets, fixing one problem often creates another

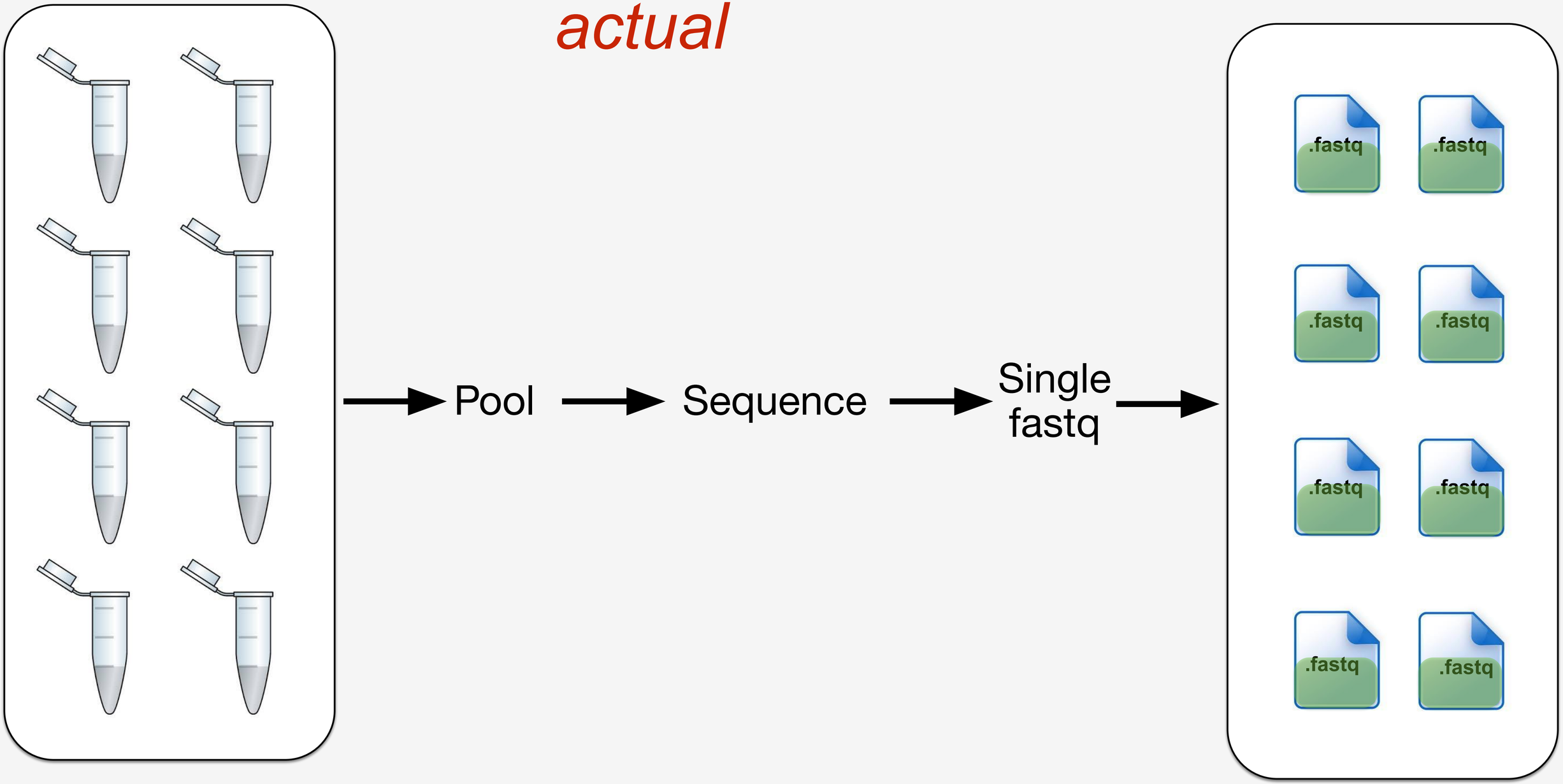


genes with lower expression have highest StdDev

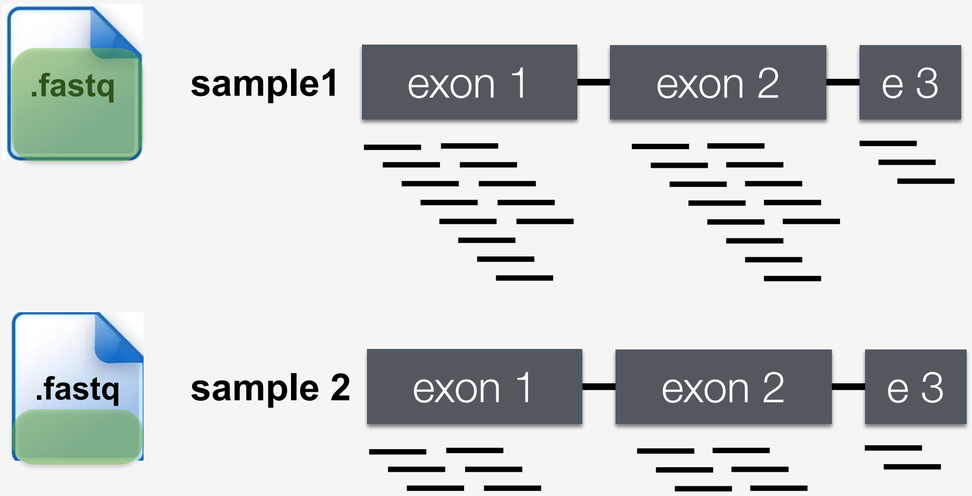
expected reads per sample

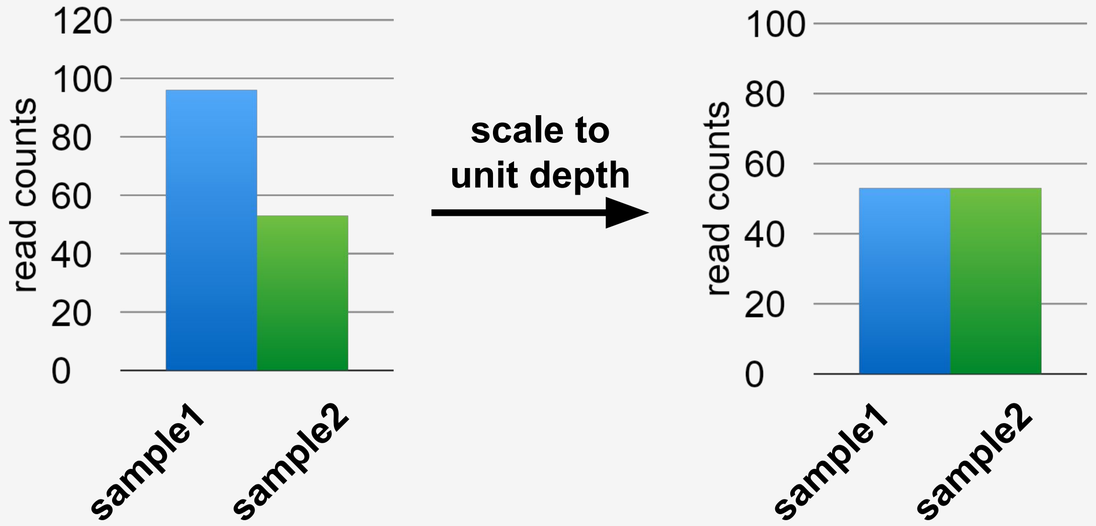


~~expected~~ actual reads per sample

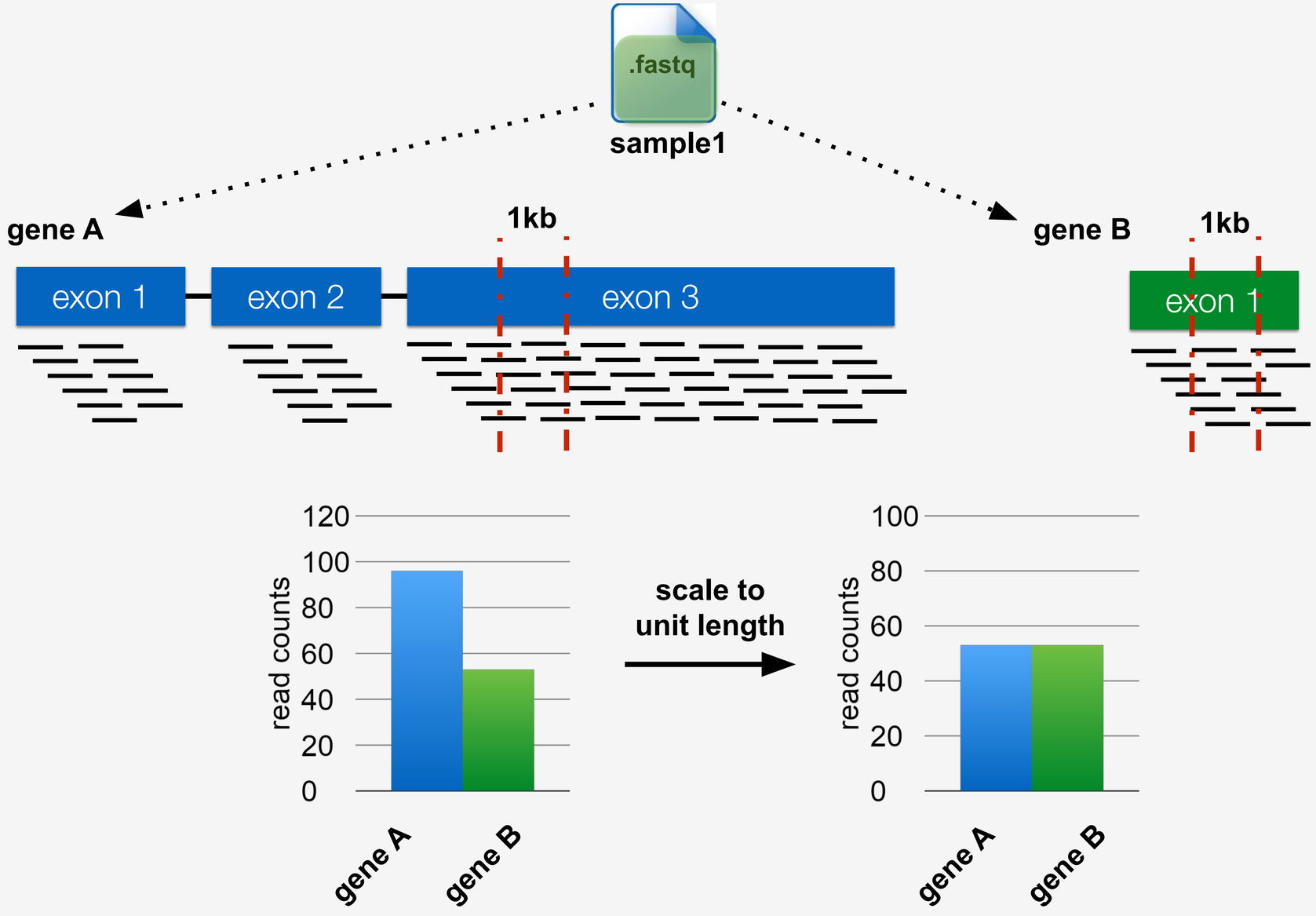


scaling units for between libraries

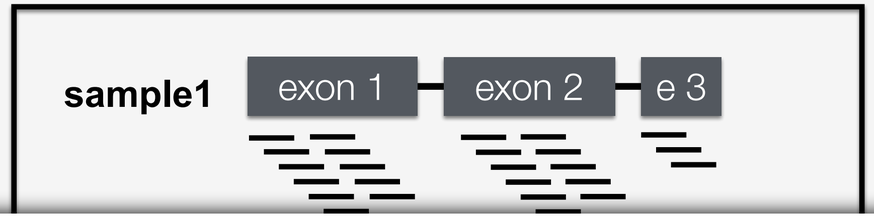




scaling units for within-sample comparisons



reads per kilobase, per million reads sequenced (RPKM)

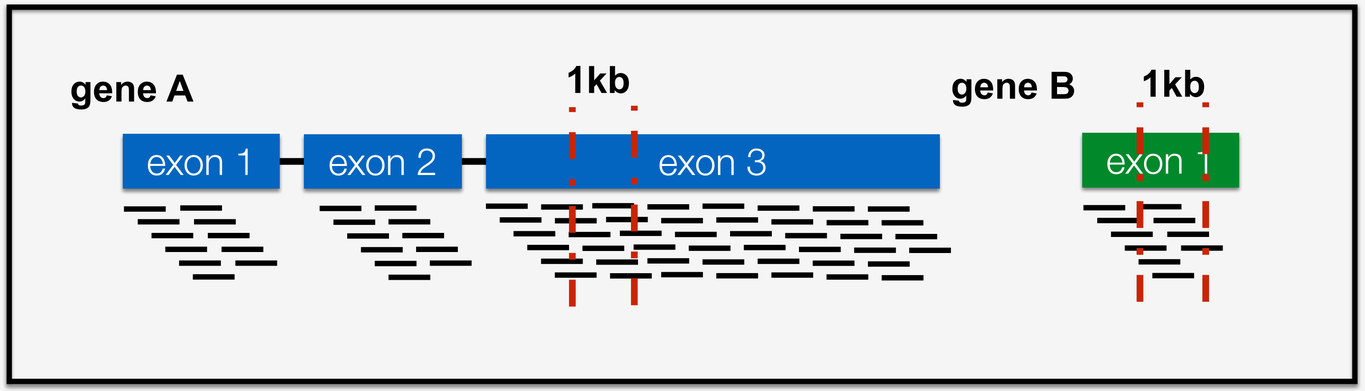


Original RPKM ref:

RPKM is really just a unit of expression for RNAseq data

+

within sample



RPKM (FPKM)

$$\text{RPKM} = \frac{\#\text{ reads mapped to genomic region}}{(\text{region length in kb})(\text{total \# reads})}$$

RPKM (FPKM)

$$\textbf{RPKM} = \frac{\textbf{\# reads mapped to genomic region}}{\textbf{(region length in kb)(total \# reads)}}$$

$$\textbf{RPKM} = \frac{\textbf{1000}}{\textbf{(5)(20000000)}} = 0.00001$$

RPKM (FPKM)

$$\textbf{RPKM} = \frac{\textbf{\# reads mapped to genomic region}}{\textbf{(region length in kb)(total \# reads)}} \times 10^6$$

$$\textbf{RPKM} = \frac{1000}{(5)(20000000)} = 0.00001$$

RPKM (FPKM)

$$\text{RPKM} = \frac{\text{\# reads mapped to genomic region}}{\text{(region length in kb)(total \# reads)}} \times 10^6$$

$$\text{RPKM} = \frac{1000}{(5)(20000000)} = 10$$

RPKM alone, is not sufficient for normalization

Table 3: Summary of comparison results for the seven normalization methods under consideration

Method	Distribution	Intra-Variance	Housekeeping	Clustering	False-positive rate
TC	–	+	+	–	–
UQ	++	++	+	++	–
Med	++	++	–	++	–
DESeq DESeq	++	++	++	++	++
TMM EdgeR	++	++	++	++	++
Q	++	–	+	++	–
RPKM	–	+	+	–	–

A '–' indicates that the method provided unsatisfactory results for the given criterion, while a '+' and '++' indicate satisfactory and very satisfactory results for the given criterion.

Diilies et al., Briefings in Bioinformatics, Sept 2012

The problem with RPKM

***read counts* from each gene**

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total reads
Sample 1	80	10	6	3	1	*100*
Sample 2	20	20	10	50	400	*500*

$$\textbf{RPKM} = \frac{\textbf{\# reads mapped to genomic region}}{\textbf{(region length in kb)(total \# reads)}} \times 10^6$$

***RPKM* each gene**

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total RPKM
Sample 1	8000	2000	2400	6000	10000	***28400***
Sample 2	400	800	800	20000	800000	***822000***

RNAseq gives relative quantification of gene expression

"All commonly used techniques to measure mRNA abundance, including qPCR, microarray signals, as well as reads per kilobase per million reads (RPKM) for RNAseq data, aim at estimating a statistic that is as closely proportional to the **relative molar concentration** as possible."

- Wagner, Theories in Biosci., 2012

“The average **relative molar concentration** for each and every sample of RNA-seq data mapped to the same genome is the same constant value.

- Wagner, Theories in Biosci., 2012

The problem with RPKM

***read counts* from each gene**

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total reads
Sample 1	80	10	6	3	1	*100*
Sample 2	20	20	10	50	400	*500*

$$\textbf{RPKM} = \frac{\textbf{\# reads mapped to genomic region}}{\textbf{(region length in kb)(total \# reads)}} \times 10^6$$

***RPKM* each gene**

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total RPKM	
Sample 1	8000	2000	2400	6000	10000	*28400*	**μ = 5680**
Sample 2	400	800	800	20000	800000	*822000*	**μ = 164400**

Fixing RPKM is easy

read count

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total reads
Sample 1	80	10	6	3	1	*100*
Sample 2	20	20	10	50	400	*500*

scale by gene length first

this becomes a normalization factor

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total RPK
Sample 1	0.8	0.2	0.24	0.6	1	*2.84*
Sample 2	0.2	0.4	0.4	10	400	*411*

Fixing RPKM is easy

$$\text{TPM} = \frac{\text{reads per Kb}}{\text{total RPK in sample}}$$

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total RPK
Sample 1	0.8	0.2	0.24	0.6	1	*2.84*
Sample 2	0.2	0.4	0.4	10	400	*411*

Fixing RPKM is easy

$$\text{TPM} = \frac{\text{reads per Kb}}{\text{total RPK in sample}}$$

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total RPK
Sample 1	0.8/2.84	0.2/2.84	0.24/2.84	0.6/2.84	1/2.84	*2.84*
Sample 2	0.2/411	0.4/411	0.4/411	10/411	400/411	*411*

Fixing RPKM is easy

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total TPM
Sample 1	0.281690141	0.070422535	0.084507042	0.211267606	0.352112676	*1*
Sample 2	0.000486618	0.000973236	0.000973236	0.0243309	0.97323601	*1*

$$\text{TPM} = \frac{\text{reads per Kb}}{\text{total RPK in sample}}$$

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total RPK
Sample 1	0.8/2.84	0.2/2.84	0.24/2.84	0.6/2.84	1/2.84	*2.84*
Sample 2	0.2/411	0.4/411	0.4/411	10/411	400/411	*411*

Fixing RPKM is easy

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total TPM	
Sample 1	281690	70423	84507	211268	352113	*1000000*	**μ = 200000**
Sample 2	487	973	973	24331	973236	*1000000*	**μ = 200000**

$$\textbf{TPM} = \frac{\textbf{reads per Kb}}{\textbf{total RPK in sample}} \times 10^6$$

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total RPK
Sample 1	0.8/2.84	0.2/2.84	0.24/2.84	0.6/2.84	1/2.84	*2.84*
Sample 2	0.2/411	0.4/411	0.4/411	10/411	400/411	*411*

Further normalization steps need to be taken before one can compare between samples

"library size scaling is too simple for many biological applications. The number of fragments expected to map to a gene is not only dependent on the expression level and length of the gene, but also the composition of the RNA population that is being sampled. Thus, if a large number of genes are unique to, or highly expressed in, one experimental condition, the sequencing 'real estate' available for the remaining genes in that sample is decreased. If not adjusted for, this sampling artifact can force the DE analysis to be skewed towards one experimental condition. "

Robinson and Oshlack, Genome Biology, 2010

Further normalization steps need to be taken before one can compare <u>between</u> samples

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total reads
Sample 1	80	10	6	3	1	*100*
Sample 2	20	20	10	50	400	*500*

Calculate RPK

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total RPK
Sample 1	0.8	0.2	0.24	0.6	1	*2.84*
Sample 2	0.2	0.4	0.4	10	400	*411*

Calculate TPM

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total RPK
Sample 1	281690	70423	84507	211268	352113	*1000000*
Sample 2	487	973	973	24331	973236	*1000000*

Differential expression is all messed up! Why?!

<u>Robinson and Oshlack, Genome Biology, 2010</u>

Further normalization steps need to be taken before one can compare <u>between</u> samples

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total reads
Sample 1	80	10	6	3	1	*100*
Sample 2	20	20	10	50	[illegible]	***100***

Calculate RPK

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total RPK
Sample 1	0.8	0.2	0.24	0.6	1	*2.84*
Sample 2	0.2	0.4	0.4	10	[illegible]	***11***

Calculate TPM

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	total RPK
Sample 1	281690	70423	84507	211268	352113	*1000000*
Sample 2	18181	36363	36363	909090	[illegible]	*1000000*

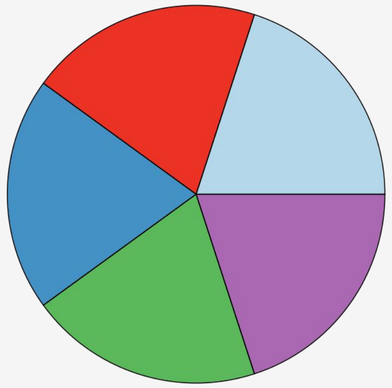
most current normalization methods attempt to find a set of genes with minimal variance across samples for normalization

<u>Robinson and Oshlack, Genome Biology, 2010</u>

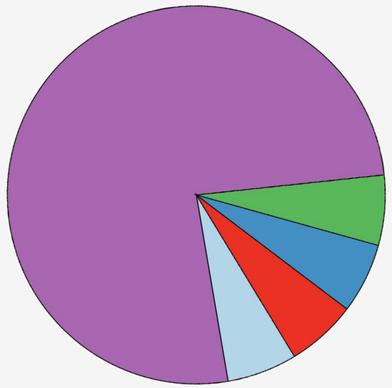
Further normalization steps need to be taken before one can compare <u>between</u> samples

	Gene A	Gene B	Gene C	Gene D	Gene E	total counts
Control	0.2	0.2	0.2	0.2	0.2	*10*
Treatment	0.06	0.06	0.06	0.06	0.76	*100*

Control



Treatment

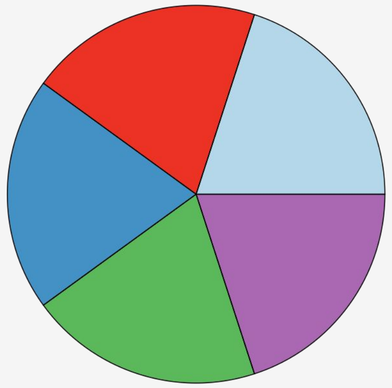


every gene appears to be differentially expressed!

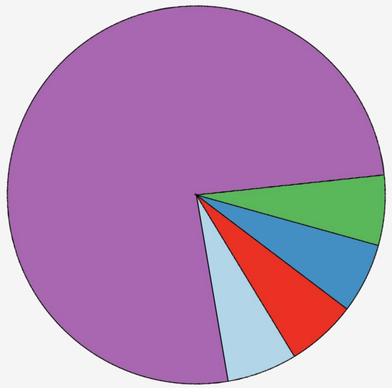
Further normalization steps need to be taken before one can compare <u>between</u> samples

	Gene A *100kb*	Gene B *50kb*	Gene C *25kb*	Gene D *5kb*	Gene E *1kb*	**Total RPK**
control	0.25	0.25	0.25	0.25	0.25	*8*
treatment	0.25	0.25	0.25	0.25	3.17	*24**

Control



Treatment



most current normalization methods attempt to find a set of genes with minimal variance across samples for normalization

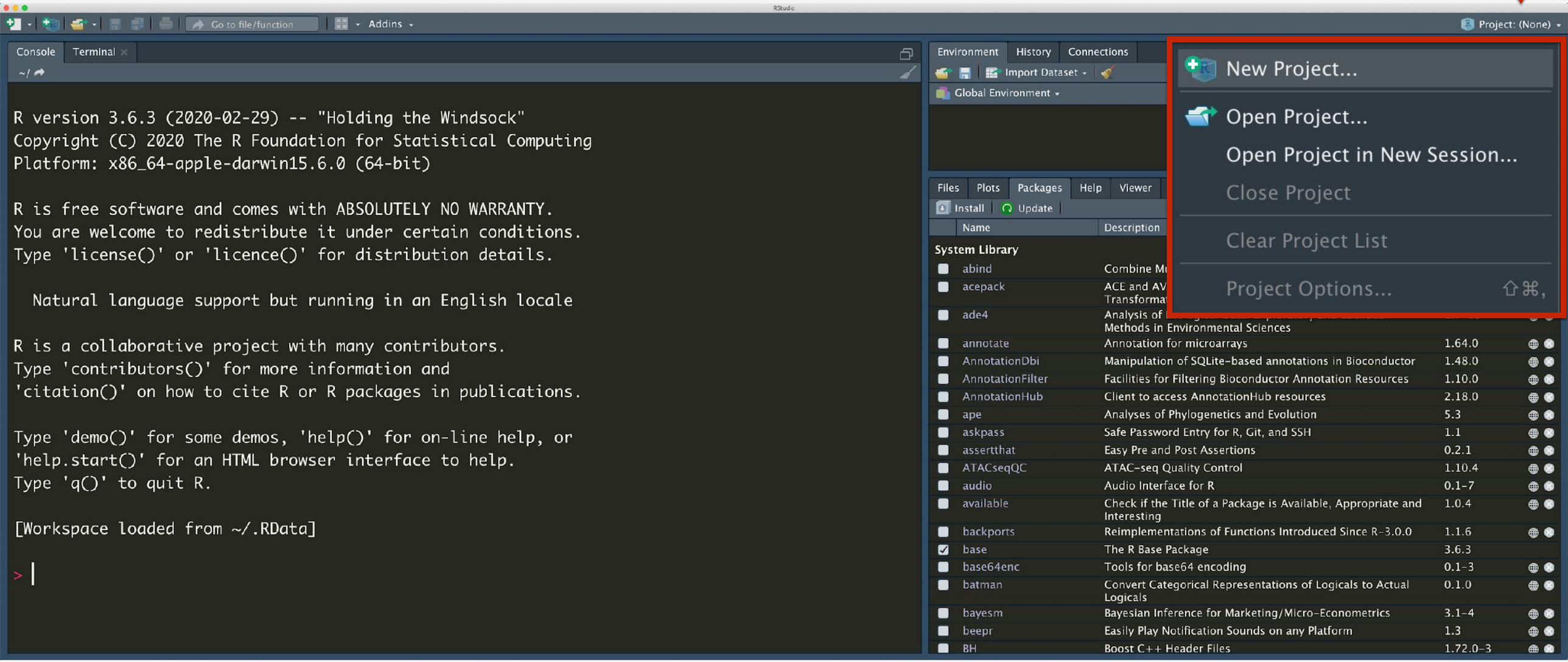
Statistical tools for normalization and differential expression analysis make certain assumptions about your data

Key assumptions

1. Most genes are not differentially expressed (implications for comparing very different treatments/conditions)
2. Approx. equivalent numbers of up and down regulated genes

End of Part 1

Setting up our analysis project

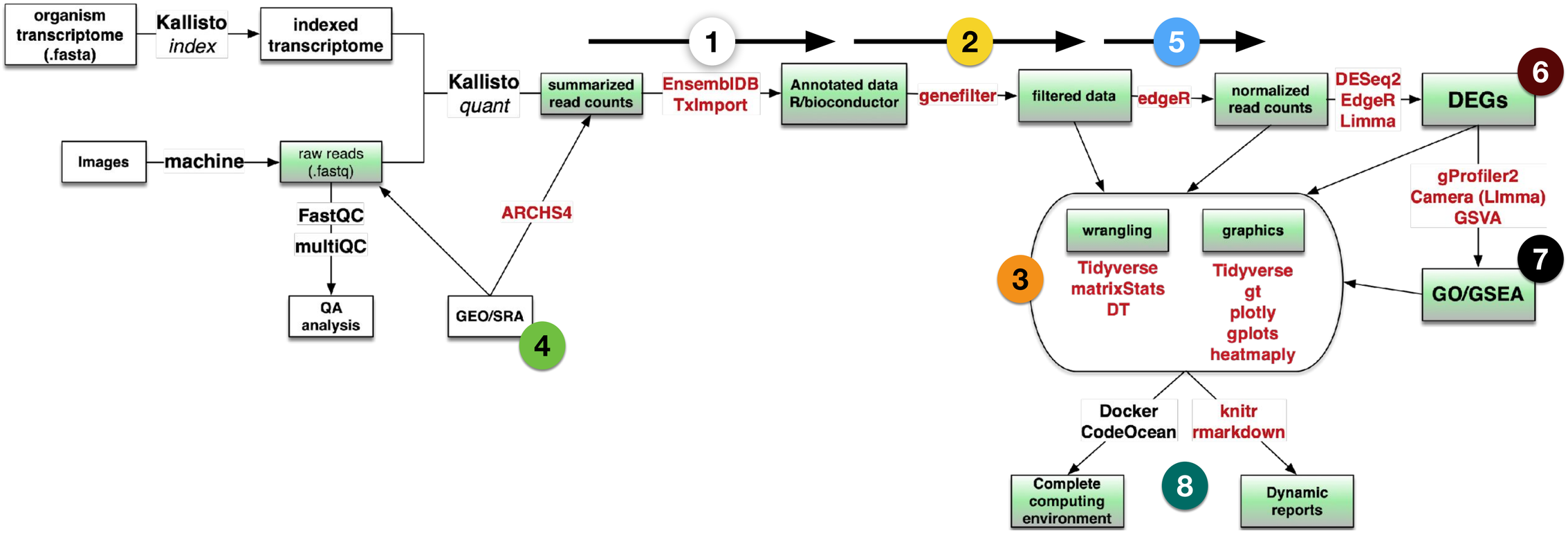


R projects

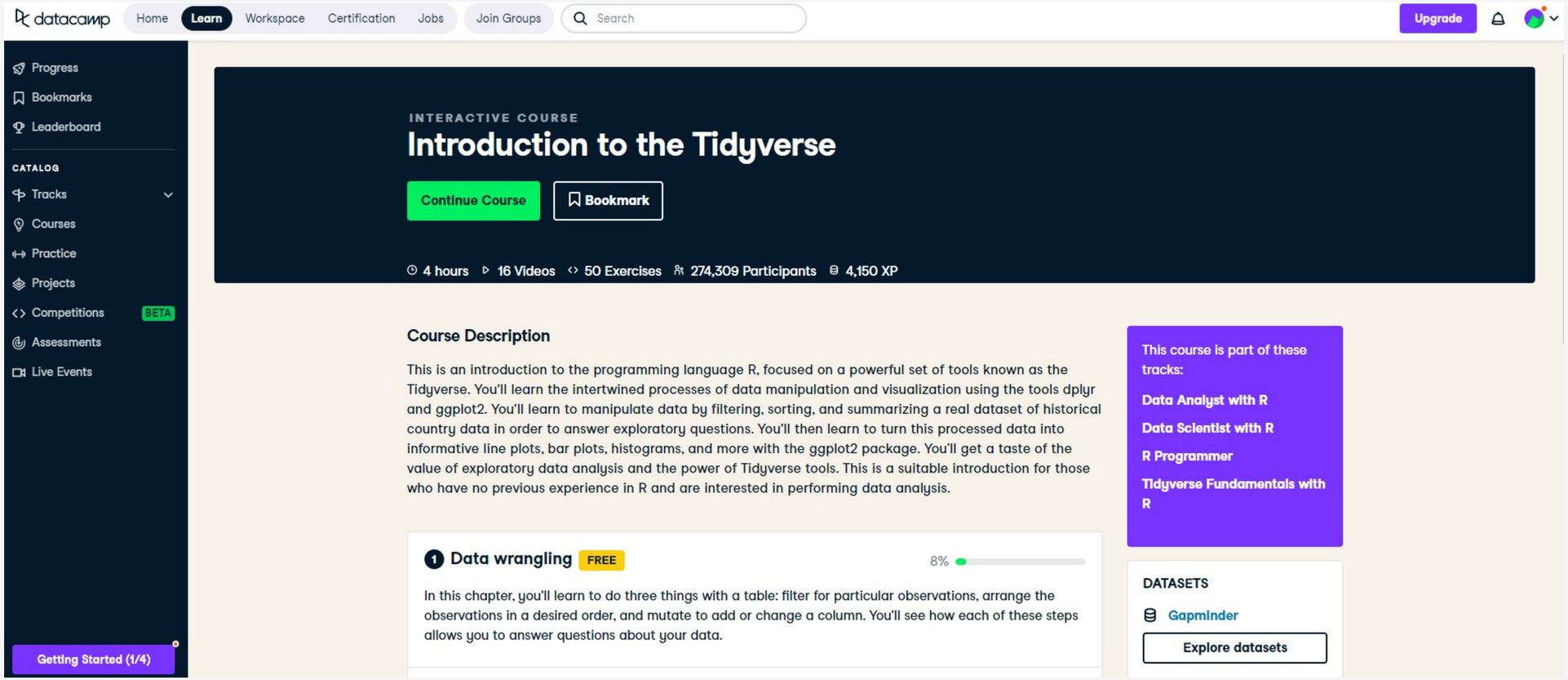
- A new R session (process) is started
- The .**Rprofile** file in the project's main directory (if any) is sourced by R
- The .**RData** file in the project's main directory is loaded (if project options indicate that it should be loaded).
- The .**Rhistory** file in the project's main directory is loaded into the RStudio History pane (and used for Console Up/Down arrow command history).
- The **current working directory** is set to the project directory.
- Previously edited **source documents** are restored into editor tabs
- Other **RStudio settings** (e.g. active tabs, splitter positions, etc.) are restored to where they were the last time the project was closed.

R script ≠ RStudio project

We'll break up our analysis into 'step' scripts



A homework assignment



https://app.datacamp.com/learn/courses/introduction-to-the-tidyverse

https://github.com/LukaIgnjatovic/DataCamp_-_Track_-_Data_Scientist_with_R_-_Course_03_-_Introduction_to_the_Tidyverse