

# Reproducible Workflows with R and Sequencing Data

## A Beginner's Guide to transcriptomic analysis

---

Adrian Sven Geissler  
7<sup>th</sup> of February 2024

A photograph of a protest or rally. In the center, a person's hand with pink-painted fingernails holds up a cardboard sign. The sign has the word "EQUALITY" written vertically along the top edge and "DIVERSITY" written horizontally across the middle. The background is a bright, overcast sky with some buildings visible.

Diverse target audience

Photo by Amy Elting on Unsplash

# Presentation takeaways

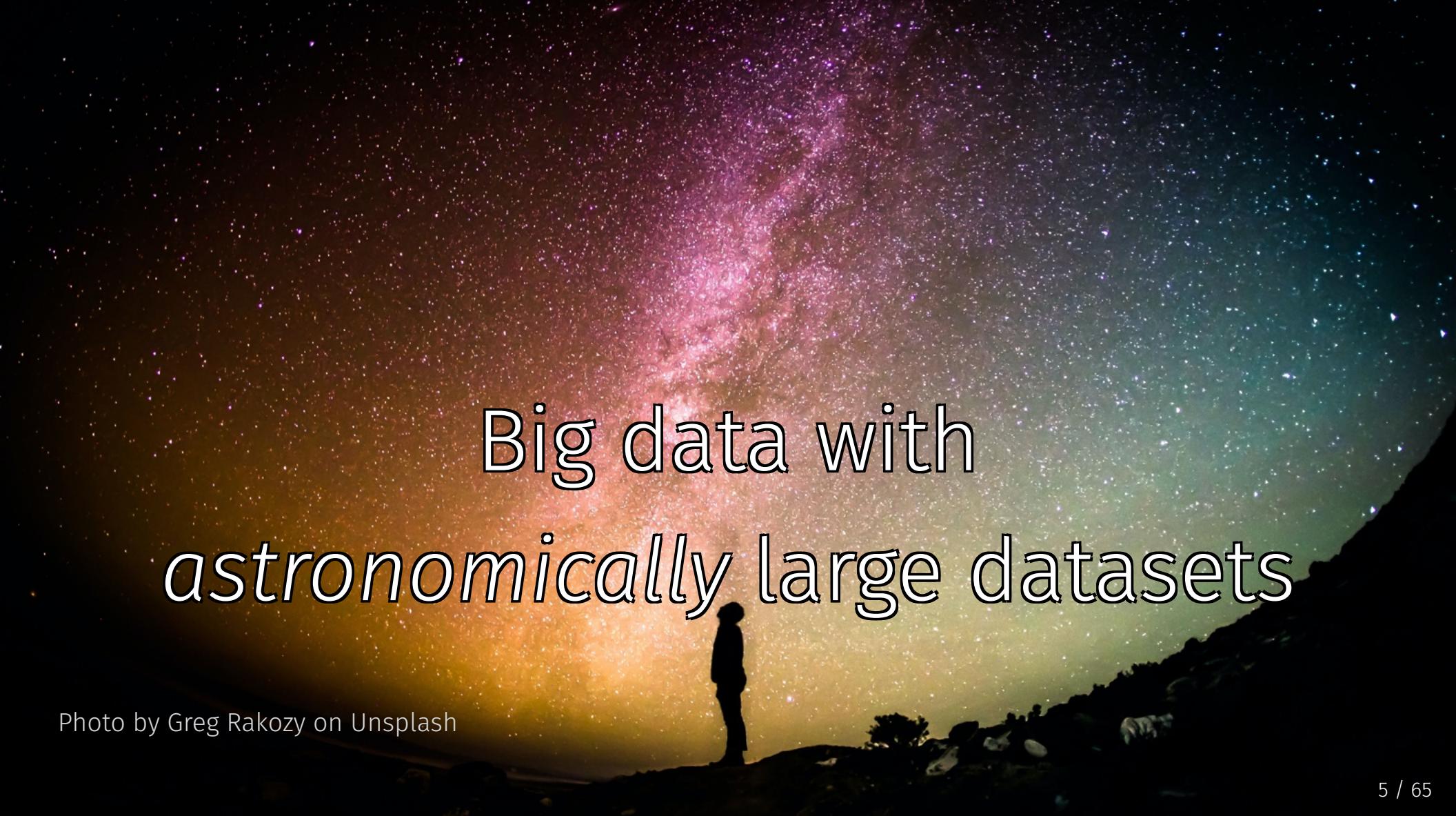
- Define "*transcriptomic*" sequencing and its relevance
- Recognize structure of sequencing data
- Outline a simple analysis pipeline with bash
- Identify concept and advantages of a Snakemake workflow  
(How I avoid using bash as much as possible)
- Connect a workflow with R analysis scripts
- Introduction to software management across platforms
- Fun with  packages and R tweaks

Slides:

<https://github.com/asgeissler/2024-CopenhagenR-Seminar>

Snakemake tutorial workflow:

<https://github.com/asgeissler/2024-Snakemake-Intro>

A photograph of a person standing on a rocky hillside at night, silhouetted against a vibrant, colorful sky filled with stars. The colors transition from deep purple at the top to orange and yellow near the horizon.

# Big data with *astronomically* large datasets

Photo by Greg Rakozy on Unsplash



**LIFE SCIENCES**

**ASTRONOMY**

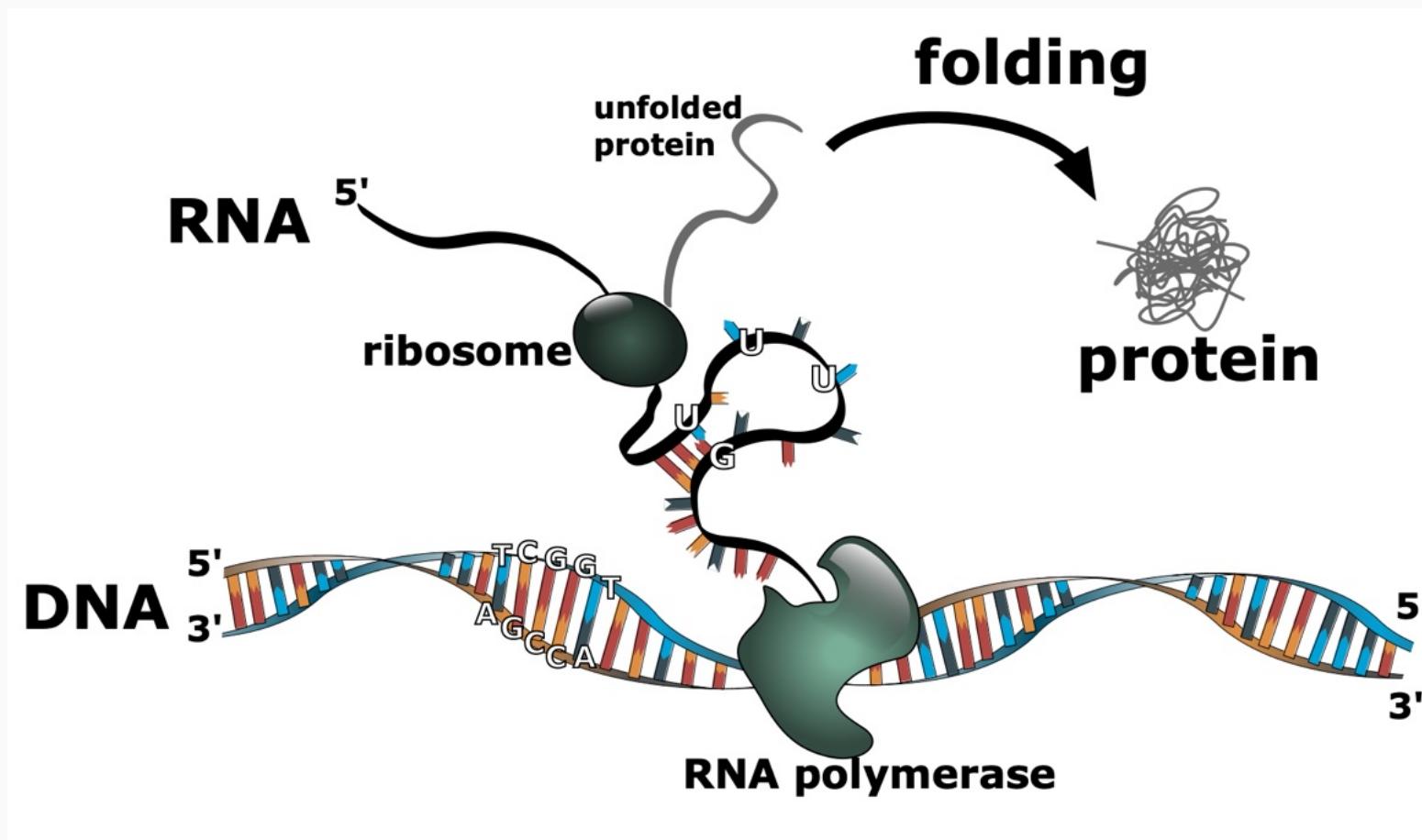
# Big Data: Astronomical or Genomical?

Data Phase	Astronomy	Twitter	YouTube	Genomics
Acquisition	25 zetta-bytes/year	0.5–15 billion tweets/year	500–900 million hours/year	1 zetta-bases/year
Storage	1 EB/year	1–17 PB/year	1–2 EB/year	2–40 EB/year
Analysis	In situ data reduction	Topic and sentiment mining	Limited requirements	Heterogeneous data and analysis
	Real-time processing	Metadata analysis		Variant calling, ~2 trillion central processing unit (CPU) hours
	Massive volumes			All-pairs genome alignments, ~10,000 trillion CPU hours
Distribution	Dedicated lines from antennae to server (600 TB/s)	Small units of distribution	Major component of modern user's bandwidth (10 MB/s)	Many small (10 MB/s) and fewer massive (10 TB/s) data movement

doi:10.1371/journal.pbio.1002195.t001

Stephens, et al. PLOS Biology 2015

# Gene expression: From genome, over RNA, to proteins



-ome suffix: A totality of some sort

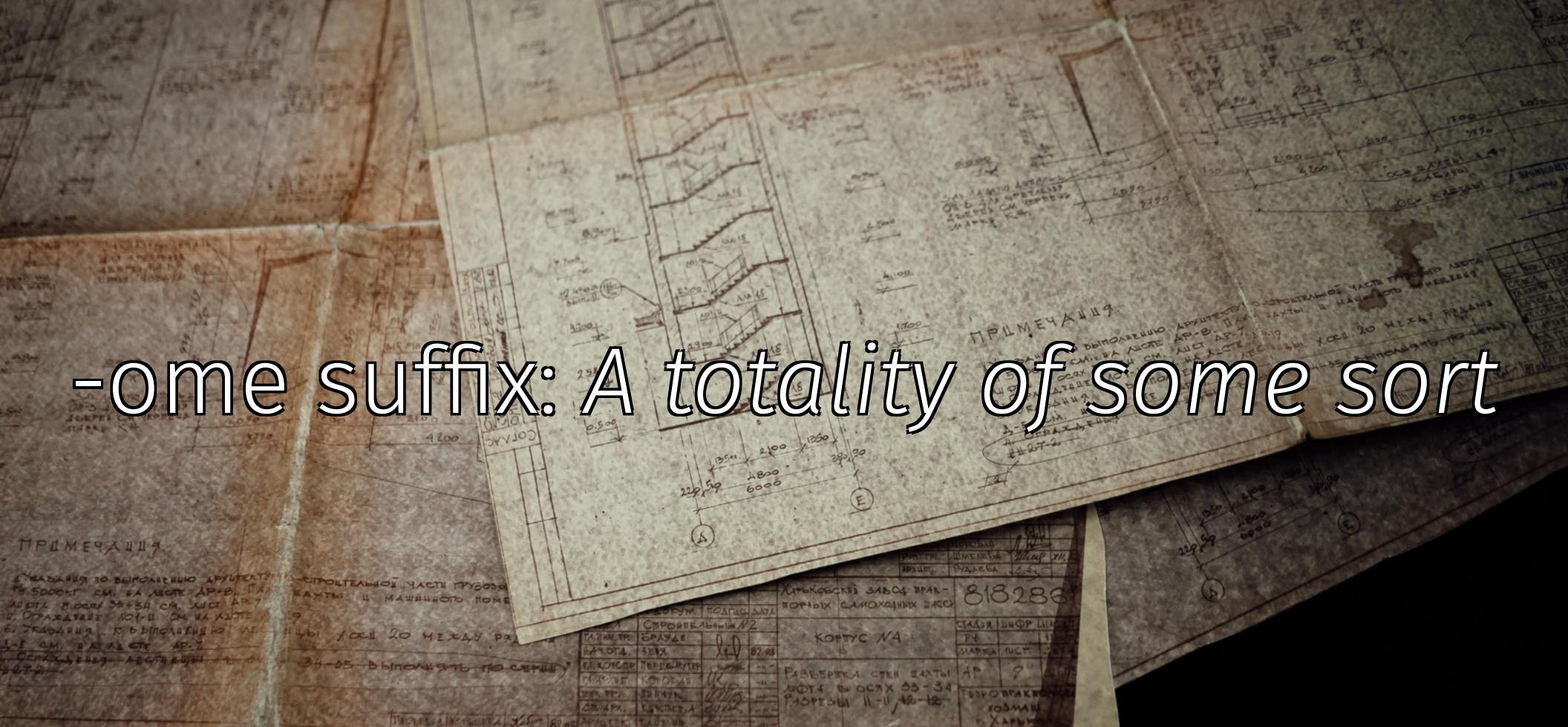
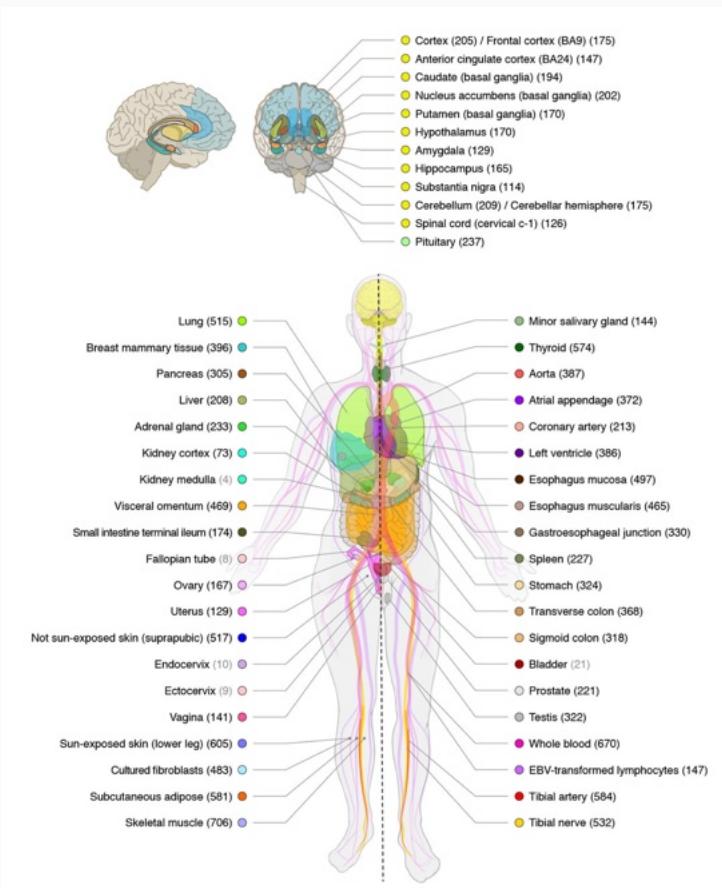


Photo by Evgeniy Surzhan on Unsplash

# Differences in gene expression drive phenotypes

- 🌐 Cell tissue
- 💡 Genome
- 🌳 Environment & 🍯 Exposure
- 💀 Disease associate factors

Figure from the GTEx Consortium, Science 2020



# Transcriptomic analysis: What is happening in the cells?

Photo by Evgeniy Surzhan on Unsplash

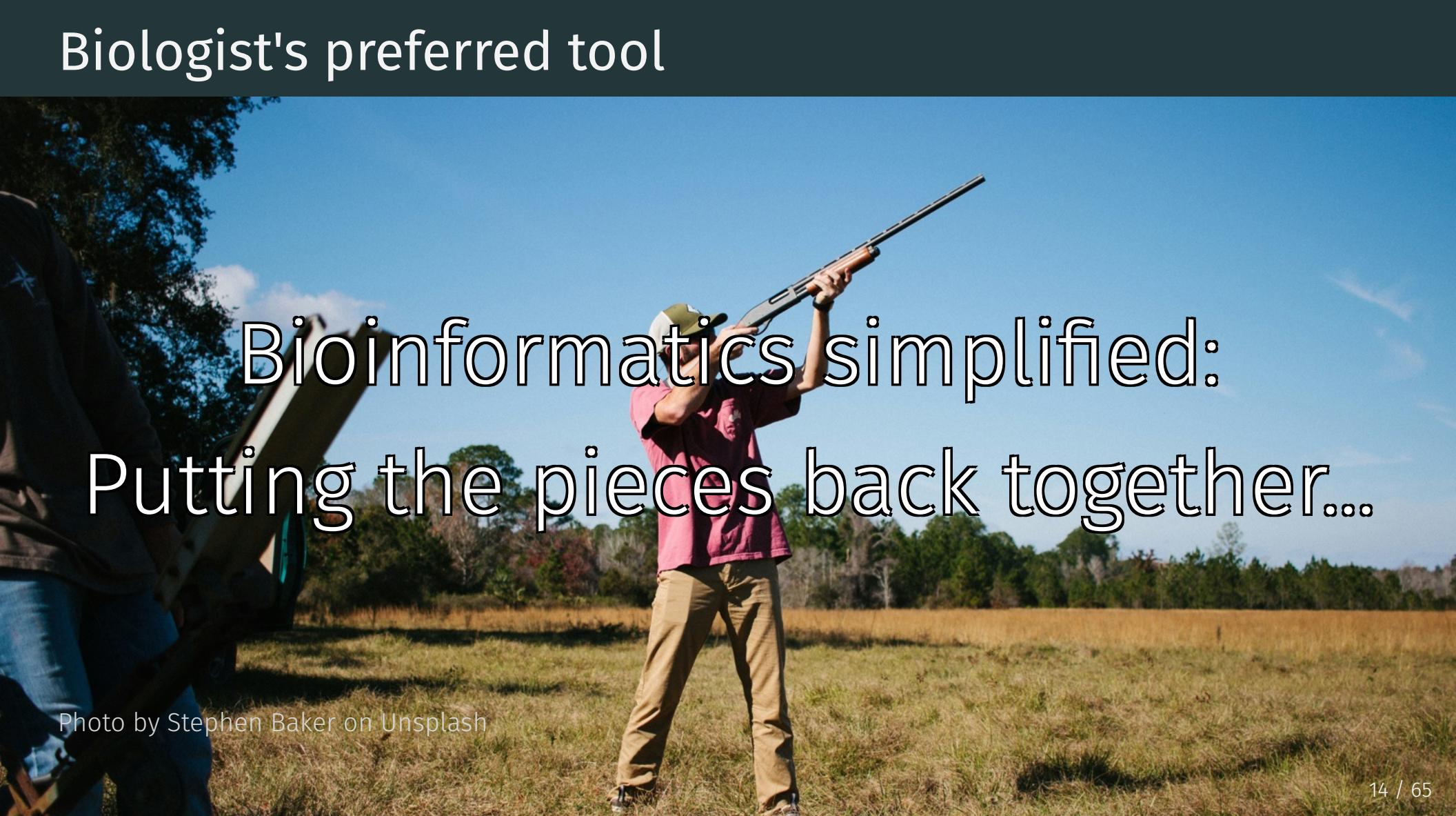


Photo by Dave Weatherall on Unsplash



Photo by Barn Images on Unsplash

# Biologist's preferred tool



Bioinformatics simplified:  
Putting the pieces back together...

Photo by Stephen Baker on Unsplash

# The pre-R fun bottleneck



Photo by Nathan Dumlao on Unsplash

Biological *high-throughput* measurements

- RNA Sequencing data
- Terabytes of **> 100** compressed txt's
- Specialized tools (bash scripts)
- Long running times

Intermediate result:

Count Matrix < **100** MB

Rows: Genes, Columns: Samples/Patients

Data exploration and analysis  
(R scripts / notebooks)

# Transcriptomic "airway" dataset from 2014

OPEN  ACCESS Freely available online

 PLOS ONE

## RNA-Seq Transcriptome Profiling Identifies *CRISPLD2* as a Glucocorticoid Responsive Gene that Modulates Cytokine Function in Airway Smooth Muscle Cells

Blanca E. Himes<sup>1,2,3\*</sup>, Xiaofeng Jiang<sup>4\*</sup>, Peter Wagner<sup>4</sup>, Ruoxi Hu<sup>4</sup>, Qiyu Wang<sup>4</sup>, Barbara Klanderman<sup>2</sup>, Reid M. Whitaker<sup>1</sup>, Qingling Duan<sup>1</sup>, Jessica Lasky-Su<sup>1</sup>, Christina Nikolos<sup>5</sup>, William Jester<sup>5</sup>, Martin Johnson<sup>5</sup>, Reynold A. Panettieri Jr.<sup>5</sup>, Kelan G. Tantisira<sup>1</sup>, Scott T. Weiss<sup>1,2</sup>, Quan Lu<sup>4\*</sup>

- Freely and openly available (without ethics board)
- RNA sequencing datasets for Asthma drug treatment vs control
- 4 human cell lines (genotypes)

# Downloading the human genome

The screenshot shows the GENCODE website interface. At the top, there's a logo with the word "GENCODE" in blue and green, accompanied by a stylized DNA helix icon. Below the logo is a navigation bar with four items: "Human", "Mouse", "How to access data", and "FAQ". The main content area is titled "HUMAN" in large capital letters. Underneath it, it says "GENCODE 44 (17.07.23)". A large, semi-transparent portrait of a young person's face is centered in the background of this section.

<https://www.gencodegenes.org/>

Copy and paste links from website:

```
$ URL="https://ftp.ebi.ac.uk/pub/\n    databases/gencode/\\n\nGencode_human/release_44/"\n\n$ wget $URL/GRCh38.primary_assembly.genome.fa.gz \\n\n-O genome.fna.gz\n\n$ wget $URL/gencode.v44.basic.annotation.gff3.gz \\n\n-O genome.gff.gz
```



806 MBytes compressed text; 3 GBytes uncompressed

```
$ gunzip -c genome.fna.gz | less  
>chr1 1  
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN  
...  
TGCTCATGAAGTGTGAGTTAATGCACTCAAATCAATGGTTGTGCACGGTTATATGAATA  
TTAGTGATTACAAAATATTATCAATAGACCTTGTCAACAATGTTATTGAAGAACTAATCA  
TCTATTGCTTATTAGGTCTTCTCCTGCCAGAACATGCGCTCCAGGTGGAGAGGTAT  
GTTGCCTTATCCGTGGCTGGATATAGAGATTCCCACACTGCCTTGCACACGAGCACTG  
CTGGGTAAATATTGTTGGCTGCAGGAAACGTGAAGGAATAGGCCCTCCAATGGGAGGA  
...  
>chr2 2  
...
```

# Gene annotations

34 MBytes compressed, 902 MBytes uncompressed

```
$ gunzip -c genome.gff.gz | less
##gff-version 3
#description: evidence-based annotation of the human genome (GRCh38), version 44 (Ensembl 110)
[ ... ]
chr1      HAVANA      gene      11869      14409      .          +          .          ID=ENSG00000290825.1;[ ... ];gene_name=DDX11L2;[ ... ]
...
...
```

Chromosome 1 has a gene at positions **11,869** to **14,409** on the **+1** strand.

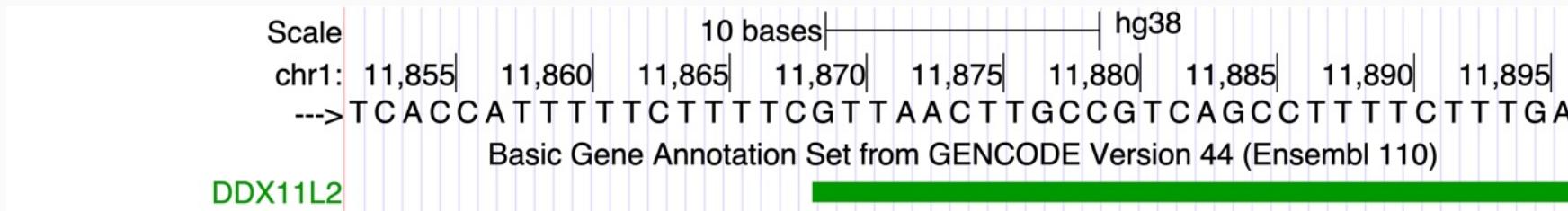


Figure from <https://genome-euro.ucsc.edu>

# Dataset access: Sequence Read Archive

From publication text: "The RNA-Seq data is available at..."

The screenshot shows the SRA study page for SRP033351. At the top, there's a banner with the NIH logo and the text "An official website of the United States government Here's how you know". Below the banner, the page header includes the SRA logo, the text "National Library of Medicine", and "National Center for Biotechnology Information". The main navigation menu includes links for "Search", "Run Browser", "Analyses", "Study", "Provisional SRA", "Documentation", and "Mirroring". The breadcrumb navigation shows "Study > SRP033351". The search bar contains the identifier "SRP033351" and a "Search" button. A help link "What can be entered in this field?" is also present. The study title is "Human Airway Smooth Muscle Transcriptome Changes in Response to Asthma Medications". The study details section includes a table with three rows: "Identifiers" (listing SRA: SRP033351, BioProject: PRJNA229998, and GEO: GSE52778), "Study Type" (Transcriptome Analysis), and "Abstract" (a detailed paragraph about the study's rationale, methods, and results). The abstract text is as follows:

Rationale: Asthma is a chronic inflammatory airway disease. The most common medications used for its treatment are  $\beta_2$ -agonists and glucocorticosteroids, and one of the primary tissues that these drugs target in the treatment of asthma is the airway smooth muscle. We used RNA-Seq to characterize the human airway smooth muscle (HASM) transcriptome at baseline and under three asthma treatment conditions. Methods: The Illumina TruSeq assay was used to prepare 75bp paired-end libraries for HASM cells from four white male donors under four treatment conditions: 1) no treatment; 2) treatment with a  $\beta_2$ -agonist (i.e. Albuterol, 1 $\mu$ M for 18h); 3) treatment with a glucocorticoid (i.e. Dexamethasone (Dex), 1 $\mu$ M for 18h); 4) simultaneous treatment with a  $\beta_2$ -agonist and glucocorticoid, and the libraries were sequenced with an Illumina Hi-Seq 2000 instrument. The Truvari Saito Tools were used to align reads to the hg19

<https://trace.ncbi.nlm.nih.gov/Traces/?view=study&acc=SRP033351>

# Downloaded sample files

- **Option 1:** Using the custom SRA Toolkit (bash...)
- **Option 2:** Clicking on website

```
$ ls  
raw-data      genome.fna.gz    genome.gff.gz  
  
$ ls -1 raw-data  
SRR1039508_1.fastq.gz  # (control sample)  
SRR1039508_2.fastq.gz  
SRR1039509_1.fastq.gz  # (asthma treated case)  
SRR1039509_2.fastq.gz  
...
```

In this dataset, the files are **3.3 – 7.3** GBytes large  
( **0.9 – 1.7** GBytes compressed)

# Paired-end sequencing data



```
> ls -1 raw-data  
SRR1039508_1.fastq.gz  
SRR1039508_2.fastq.gz  
...
```

The data contains the "*front*" and the "*back*" of the sequenced read

Photo by Taylor Smith on Unsplash

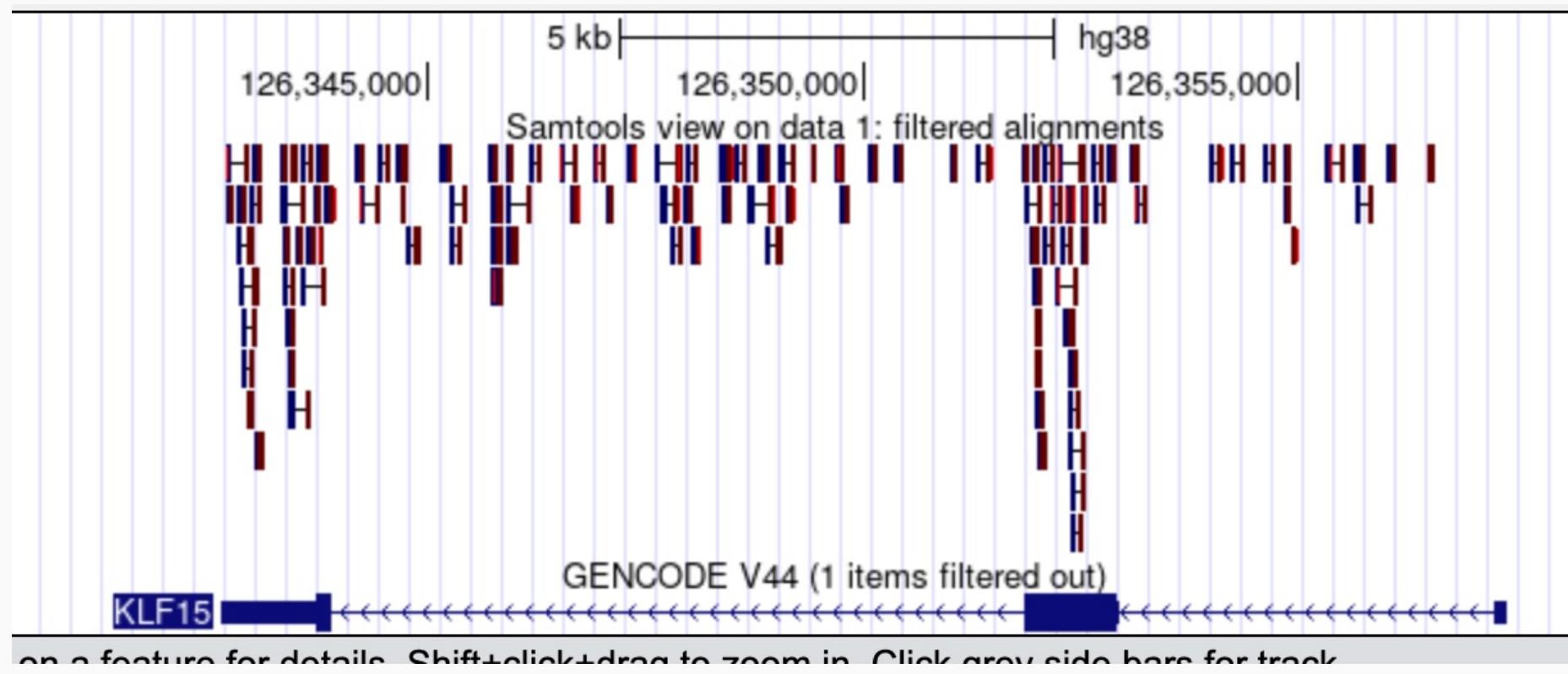
# Lots of compressed text files

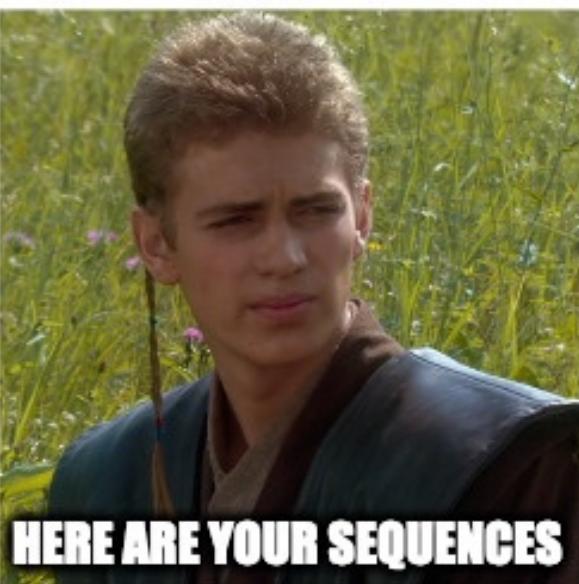
```
> gunzip -c data/SRR1039508_1.fastq.gz | head
@SRR1039508.1 HWI-ST177:290:C0TECACXX:1:1101:1225:2130 length=63
CATTGCTGATACCAANNNNNNNGATTCCCTCAAGGTCTTCCTCCCTTACGGAATTACA
+SRR1039508.1 HWI-ST177:290:C0TECACXX:1:1101:1225:2130 length=63
HJJJJJJJJJJJJJJJJJ#####00?GHIJJJJJJJJJJJJJJJJJJJJJJJJJJJJHFFFD
...
...
```

That look like one of those **ACTG** genome thingies!



# Goal: Number of reads mapping to a gene





**HERE ARE YOUR SEQUENCES**



**OH JUST  
SIMPLE TEXT LOOKUP RIGHT?**



[imgflip.com](http://imgflip.com)



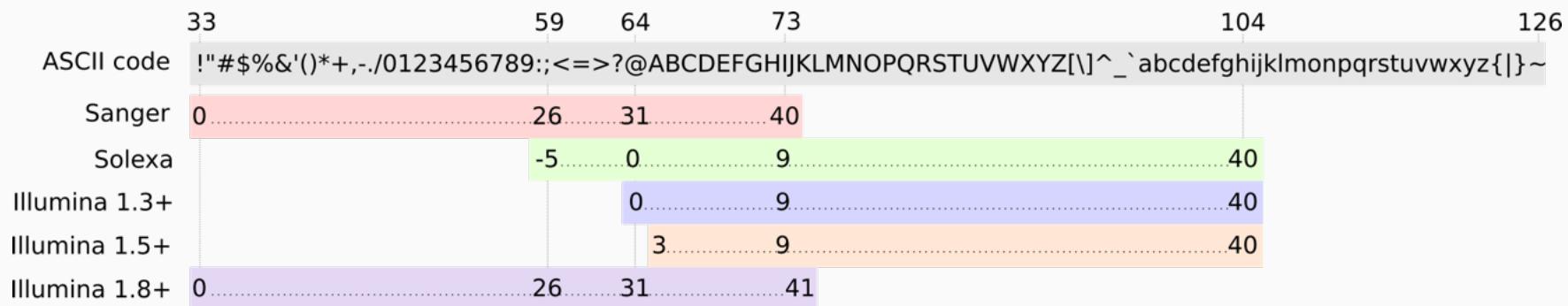
**RIGHT?**

# Sequencing quality filtering is needed

$-10 \cdot \log_{10}(\text{Probability of sequencing error})$

Encoded as single, *printable ASCII* character

(details depend on the sequencing platform manufacturer)



<https://usegalaxy.be/training-material//topics/metagenomics/tutorials/metatranscriptomics/slides-plain.html>

# Efficient sequence mapping algorithm is needed

- Short reads: < 100 nucleotides (characters)
- This dataset: 13 – 34 million read pairs, per sample
- Human genome: > 3.1 billion base pairs (characters)

~~Naive string lookup~~



# Efficient sequence mapping algorithms

1. Search via a phonebook-like index structure  
(e.g., Burrows–Wheeler Transformation)
2. Approximate via sequence spectrum  
( $k$ -mer frequency)

Suggested reading:

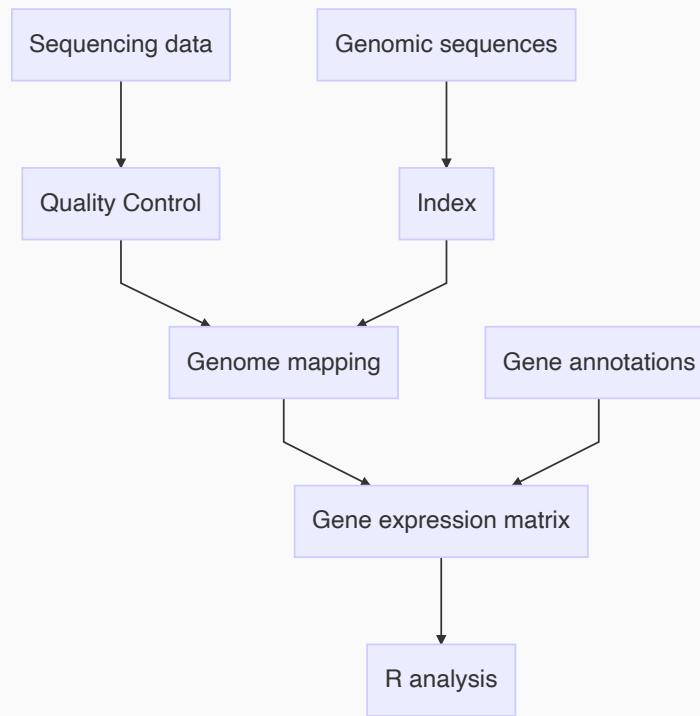
Li, Durbin. Bioinformatics 2009 <https://doi.org/10.1093/bioinformatics/btp324>

Patro, et al. Nature Methods 2017 <https://doi.org/10.1038/nmeth.4197>



Photo by Johnny Briggs  
on Unsplash

# Outline processing pipeline



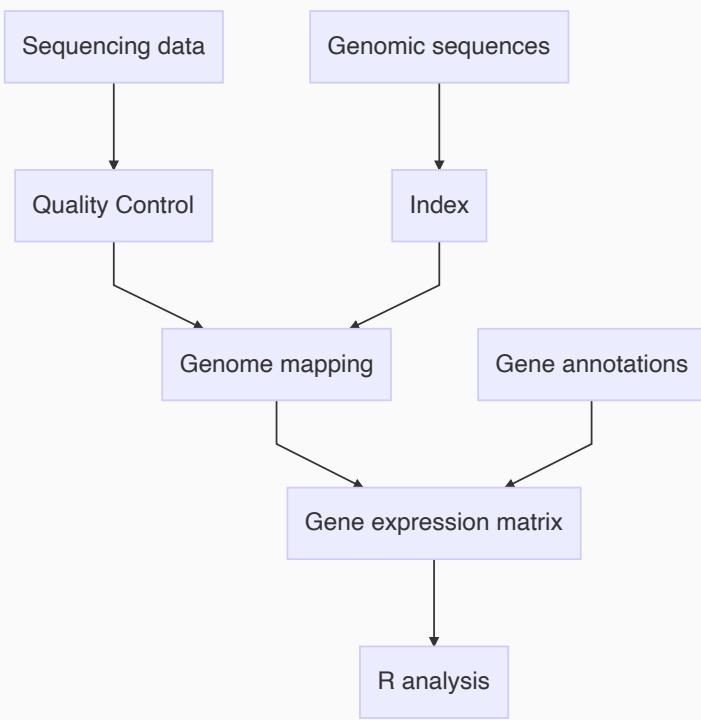
# Bash programming primer

```
qcTool raw-data/sample1.fastq.gz > clean-data/sample1.fastq.gz  
qcTool raw-data/sample2.fastq.gz > clean-data/sample2.fastq.gz  
qcTool raw-data/sample3.fastq.gz > clean-data/sample3.fastq.gz  
qcTool raw-data/sample4.fastq.gz > clean-data/sample4.fastq.gz  
...
```

Slightly better solution

```
for i in raw-data/*fastq.gz ; do  
    filename=$(basename $i)  
    qcTool $i > clean-data/$filename  
done
```

# Conceptual bash pipeline



A file `concept-pipeline.bash`

```
for i in raw-data/*.fastq.gz ; do  
    filename=$(basename $i)  
    qcTool $i > clean-data/$filename  
done  
  
makeIndex genome.fna.gz index  
  
for i in clean-data/*.fastq.gz ; do  
    filename=$(basename $i .fastq.gz)  
    mapTool index $i > mapping/$filename.bam  
done  
  
countExpression annotation.gff.gz mapping/* \  
    > matrix.txt
```

```
ngz595@SUN1027188:~/Local/2024-Snakemake-Intro/1-Concept-Bash-Workflow
```

```
ls
```

```
concept-pipeline.sh
```

```
bash concept-pipeline.sh
```

```
Created some empty files to simulate what the input might look like:
```

```
concept-pipeline.sh      genome.fna.gz      genome.gff.gz
```

```
raw-data:
```

```
sample1.fastq.gz      sample2.fastq.gz      sample3.fastq.gz      sample4.fastq.gz
```

```
Run of the conceptual pipeline via bash:
```

```
qcTool raw-data/sample1.fastq.gz > clean-data/sample1.fastq.gz
qcTool raw-data/sample2.fastq.gz > clean-data/sample2.fastq.gz
qcTool raw-data/sample3.fastq.gz > clean-data/sample3.fastq.gz
qcTool raw-data/sample4.fastq.gz > clean-data/sample4.fastq.gz
makeIndex genome.fna.gz index
mapTool index clean-data/sample1.fastq.gz > mapping/sample1.bam
mapTool index clean-data/sample2.fastq.gz > mapping/sample2.bam
mapTool index clean-data/sample3.fastq.gz > mapping/sample3.bam
mapTool index clean-data/sample4.fastq.gz > mapping/sample4.bam
countExpression annotation.gff.gz mapping/* > matrix.txt
```

```
After the run, the files are now:
```

```
concept-pipeline.sh      genome.fna.gz      genome.gff.gz      index      matrix.txt
```

```
clean-data:
```

```
sample1.fastq.gz      sample2.fastq.gz      sample3.fastq.gz      sample4.fastq.gz
```

```
mapping:
```

```
sample1.bam      sample2.bam      sample3.bam      sample4.bam
```

```
raw-data:
```

```
sample1.fastq.gz      sample2.fastq.gz      sample3.fastq.gz      sample4.fastq.gz
```

```
snakemake64 at 14:24:49
```

```
snakemake64 at 14:25:18
```

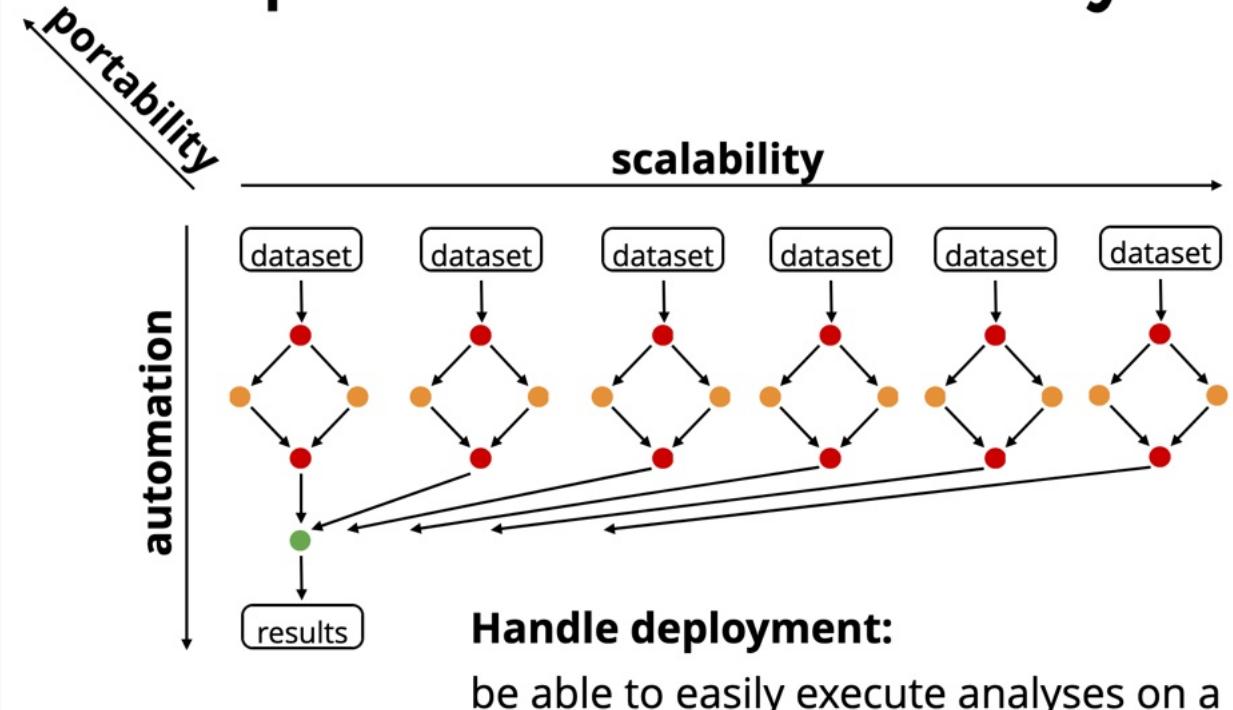
```
snakemake64 at 14:25:20
```

# Bash concerns

- Adding new samples? Re-run pipeline & over-write files?
- Running steps in parallel?
- Consistency when changing parameters?
- Software management?



## Reproducible data analysis



# Towards Snakemake 🍷 "cooking" rules

```
$ cat Snakefile
```

```
rule make_my_index:  
    input:  
        'genome.fna.gz'  
    output:  
        'index'  
    shell:  
        """  
            makeIndex {input} {output}  
        """
```

```
snakemake --cores all make_my_index
```

or

```
snakemake --cores all index
```

# Wildcard placeholders

```
rule qc_cleaning:  
    input:  
        'raw-data/{sample}.fastq.gz'  
    output:  
        'clean-data/{sample}.fastq.gz'  
    shell:  
        """  
            qcTool {input} > {output}  
        """
```

```
$ snakemake --cores all clean-data/sample1.fastq.gz  
Select jobs to execute ...  
rule qc_cleaning:  
    input: raw-data/sample1.fastq.gz  
    output: clean-data/sample1.fastq.gz  
    jobid: 1  
    reason: Missing output files: clean-data/sample1.fastq.gz  
    wildcards: sample=sample1
```

# Rules with list and named parameters

```
xs = ['sample1', 'sample2', 'sample3', 'sample4']

rule count_expression:
    input:
        genome = 'genome.gff.gz',
        mapping = expand('mapping/{sample}.bam', sample = xs)
    output:
        'matrix.txt'
    shell:
        """
        echo "I am the named genome input: {input.genome}"
        echo "We are the input mappings: {input.mapping}"
        echo "I am the first mapping: {input.mapping[0]}"

        countExpression {input} > {output}
        """

```

```
$ snakemake --cores all matrix.txt
```

```
...
```

```
I am the named genome input: genome.gff.gz
```

```
We are the input mappings: mapping/sample1.bam mapping/sample2.bam mapping/sample3.bam ...
```

```
I am the first mapping: mapping/sample1.bam
```

```
$ qcTool  
bash: qcTool: command not found
```

What are the actual tools, and how to use them?

Thank you for this short intro to bash...

But: Can we do with less bash coding, please?



## Wrappers and Meta-Wrappers

### Wrappers

ADAPTERREMOVAL

ARRIBA

ART

ASSEMBLY-STATS

BAMTOOLS

BARRNAP

RAZAM

```
rule star_index:  
    input:  
        fasta="{genome}.fasta",  
    output:  
        directory("{genome}"),  
    message:  
        "Testing STAR index"  
    threads: 1  
    params:  
        extra="",  
    log:  
        "logs/star_index_{genome}.log",  
    wrapper:  
        "v3.3.5/bio/star/index"
```

<https://snakemake-wrappers.readthedocs.io/en/stable/wrappers/star/index.html>



# Automatic dependencies resolution

When running with

```
snakemake --use-conda
```

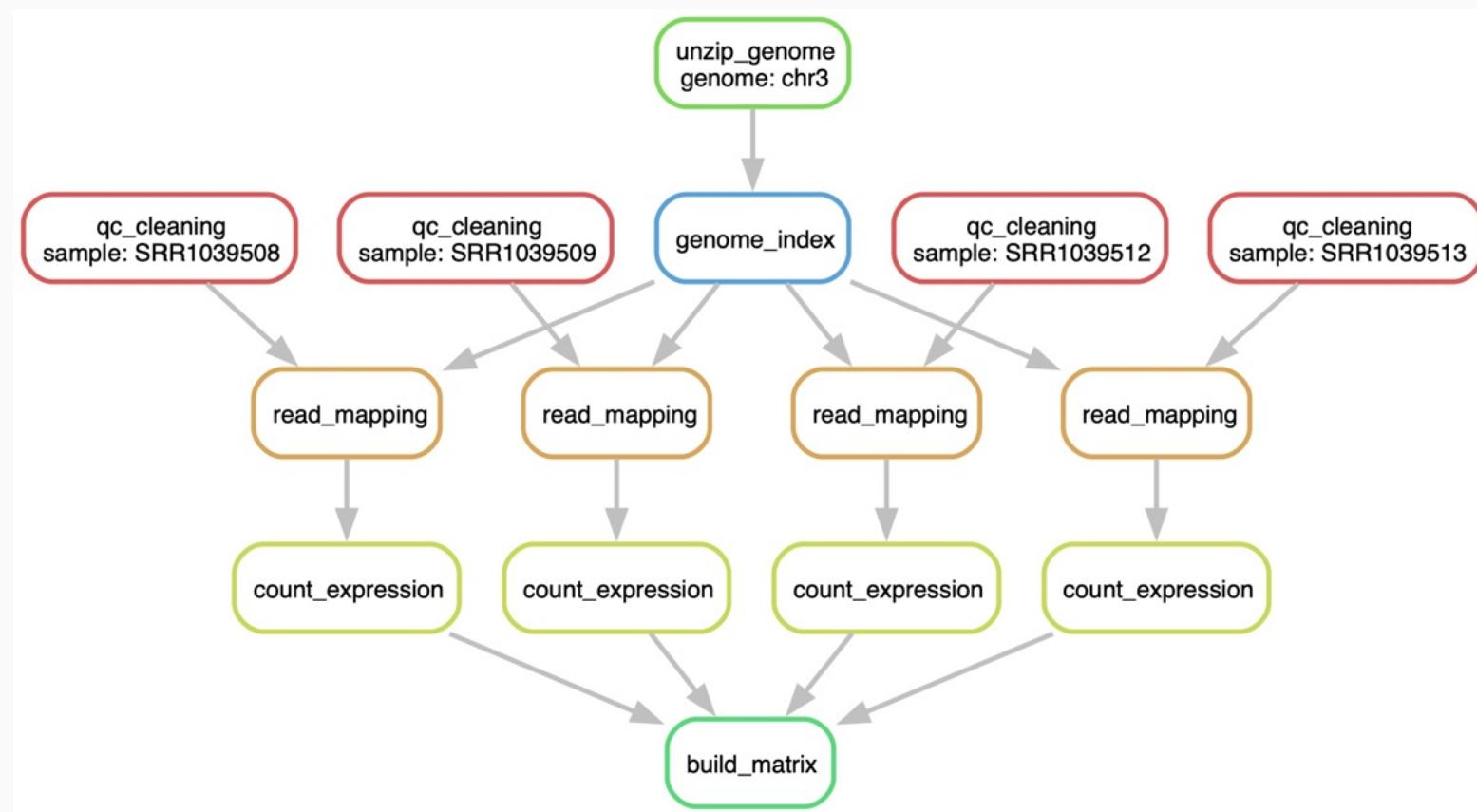
the software dependencies will be automatically deployed into an isolated environment before execution.

## Software dependencies

- star=2.7.11a

<https://snakemake-wrappers.readthedocs.io/en/stable/wrappers/star/index.html>

# Tutorial workflow



# Tutorial workflow

```
$ git clone https://github.com/asgeissler/2024-Snakemake-Intro/  
$ cd 3-Tutorial-Workflow  
$ snakemake --cores all --use-conda --conda-frontend=mamba \  
gene-expression-matrix.tsv
```

Should run in < 15 min on a notebook

```
$ head -n 3 gene-expression-matrix.tsv  
Geneid      Chr Start End      Strand Length SRR1039508 SRR1039509 SRR1039512 SRR1039513  
ENSG00000223587.2 chr3 11745 24849      +    13105  0          1          0          0  
ENSG00000224918.1 chr3 53348 54346      -    999   0          0          0          0  
  
$ grep ENSG00000163884.4 gene-expression-matrix.tsv  
ENSG00000163884.4  chr3 126342635 126357408 - 14774      220          3384        128        1518
```

# The end of the *pre-R* fun bottleneck



Intermediate result:  
Count Matrix < 100 MB

Rows: Genes, Columns: Samples/Patients

# Data and software from Bioconductor

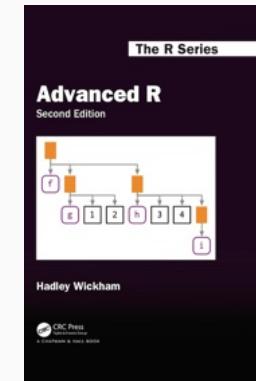


<https://bioconductor.org/>

"The mission of the Bioconductor project is [...] **open source** software [...] rigorous and **reproducible** analysis [...] welcoming **community** of developers and data scientists."

*"The Bioconductor community is a long-term user of **S4** and has produced much of the best material about its **effective use**. "*

— Hadley Wickham



# {airway}



```
install.packages("BiocManager")
BiocManager::install("airway")

library(airway)
data(airway)
airway

## class: RangedSummarizedExperiment
## dim: 63677 8
## metadata(1): ''
## assays(1): counts
## rownames(63677): ENSG0000000003 ENSG0000000005 ... ENSG00000273492
##   ENSG00000273493
## rowData names(10): gene_id gene_name ... seq_coord_system symbol
## colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
## colData names(9): SampleName cell ... Sample BioSample
```

# {airway}

Memo R pipes: `g(f(x))` is `x ▷ f() ▷ g()`  
(`▷` ligature, typed `| >`)

```
airway ▷  
assay() ▷  
as.matrix() ▷  
_[1:5, 1:5]  
  
##          SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516  
## ENSG000000000003      679       448       873       408      1138  
## ENSG000000000005       0         0         0         0         0  
## ENSG00000000419      467       515       621       365      587  
## ENSG00000000457      260       211       263       164      245  
## ENSG00000000460       60        55        40        35       78
```

The full matrix of gene (row) expression levels per sample (columns)  
from the *pre-R* fun step.

# Sample meta information

colData: Column meta information data

```
airway %> colData() %> as_tibble() %> head(n = 3) %> kable()
```

SampleName	cell	dex	albut	Run	avgLength	Experiment	Sample	BioSample
GSM1275862	N61311	untrt	untrt	SRR1039508	126	SRX384345	SRS508568	SAMN02422669
GSM1275863	N61311	trt	untrt	SRR1039509	126	SRX384346	SRS508567	SAMN02422675
GSM1275866	N052611	untrt	untrt	SRR1039512	126	SRX384349	SRS508571	SAMN02422678

Asthma drug treated (trt) or control (untrt)

kable() from {knitr}  makes pretty tables in these slides

S4 class: colData(airway) accessed the object's slot airway@colData



# Our Genes

```
airway ▷  
rowData() ▷ as_tibble() ▷  
select(gene_id, gene_name, chromosome = seq_name,  
       gene_seq_start, gene_seq_end, seq_strand) ▷  
filter(chromosome = 'X') ▷  
arrange(gene_seq_start) ▷  
mutate_if(is.numeric, prettyNum, big.mark = ',') ▷  
head(n = 3) ▷ kable()
```

gene_id	gene_name	chromosome	gene_seq_start	gene_seq_end	seq_strand
ENSG00000228572	LL0YNC03-29C1.1	X	170,410	171,758	1
ENSG00000182378	PLCXD1	X	192,989	220,023	1
ENSG00000178605	GTPBP6	X	220,025	230,886	-1

tibbles are the better dataframes {tidyverse} 

# Tidy data exploration

```
# One dexamethasone drug treated library  
my.data <-  
airway %>  
assay() %>  
as_tibble(rownames = 'gene_id') %>  
select(gene_id, expression = SRR1039509)  
my.data %> head(n = 3) %> kable()
```

gene_id	expression
ENSG000000000003	448
ENSG000000000005	0
ENSG00000000419	515

```
my.genes <-  
airway %>  
rowData() %>  
as_tibble() %>  
select(gene_id, gene_name, chromosome = sec  
      gene_seq_start, gene_seq_end, seq_st  
my.genes %> str()  
  
## #> #> #> #> #> #>
```

```
## tibble [63,677 × 6] (S3: tbl_df/tbl/data.frame)  
## #> $ gene_id : chr [1:63677] "ENSG000000000003"  
## #> $ gene_name : chr [1:63677] "TSPAN6" "TNMD3L"  
## #> $ chromosome : chr [1:63677] "X" "X" "20" "  
## #> $ gene_seq_start: int [1:63677] 99883667 99839839  
## #> $ gene_seq_end : int [1:63677] 99894988 99854985  
## #> $ seq_strand : int [1:63677] -1 1 -1 -1 1 -1
```

# Use the join!

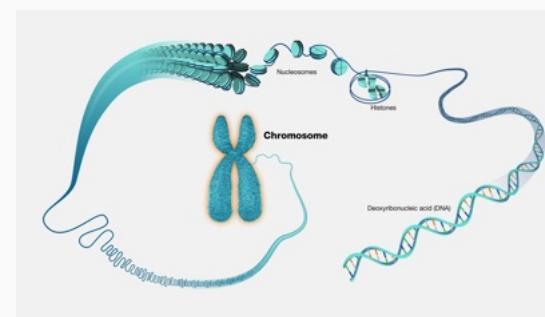
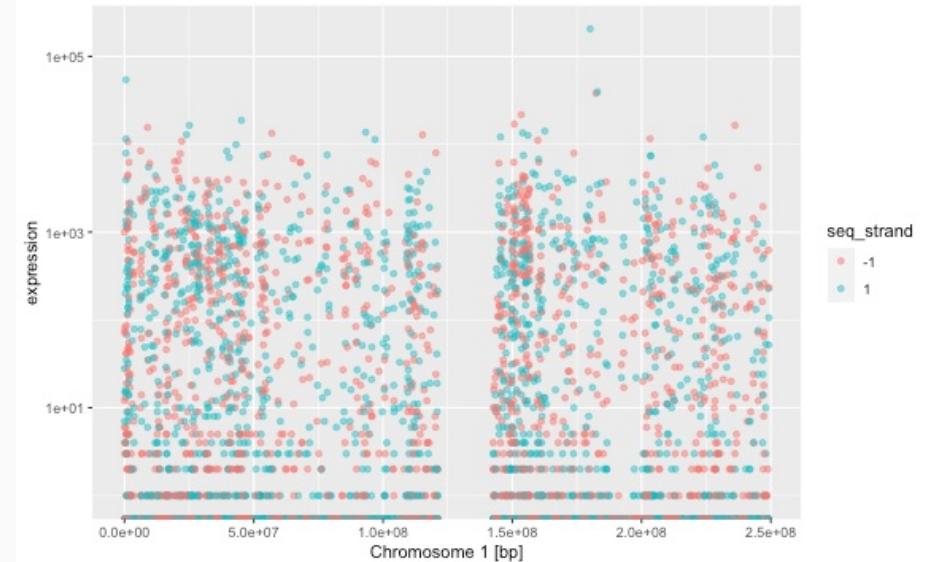
```
my.genes.data <- full_join(  
  my.genes,  
  my.data,  
  by = 'gene_id'  
)
```



gene_id	gene_name	chromosome	gene_seq_start	gene_seq_end	seq_strand	expression
ENSG000000000003	TSPAN6	X	99883667	99894988	-1	448
ENSG000000000005	TNMD	X	99839799	99854882	1	0
ENSG00000000419	DPM1	20	49551404	49575092	-1	515

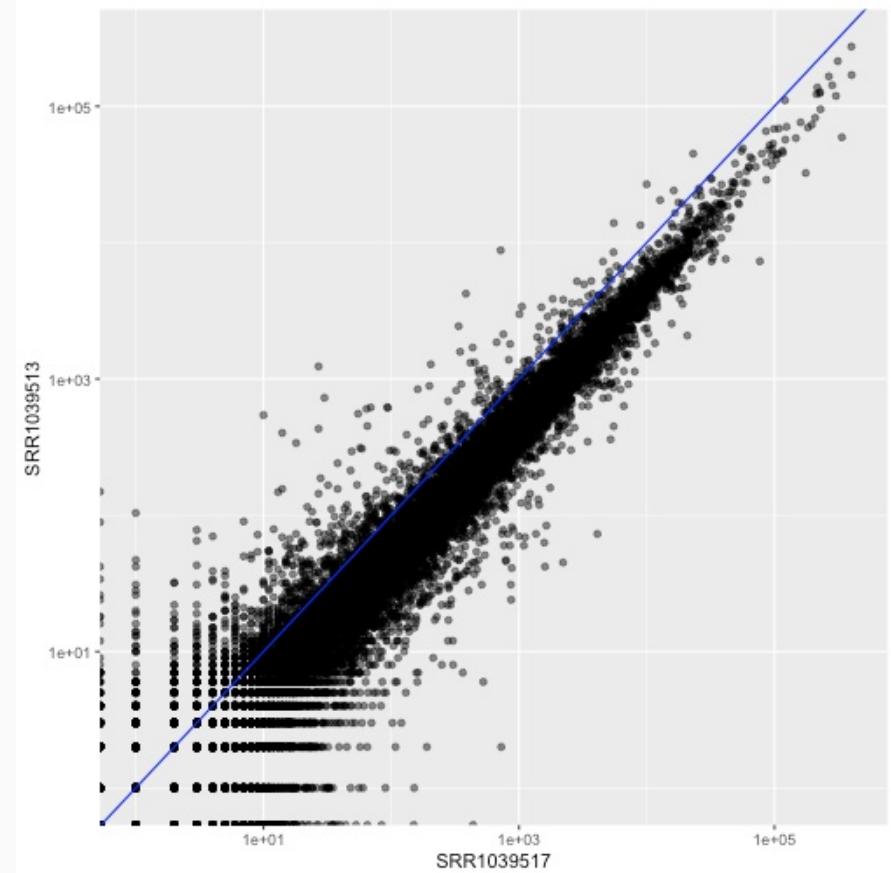
# A first genomic expression figure with {ggplot2}

```
plot1 <-  
my.genes.data %>  
filter(chromosome = '1') %>  
mutate_at('seq_strand',  
         as.character) %>  
ggplot(aes(  
  x = gene_seq_start,  
  y = expression,  
  color = seq_strand),  
) +  
geom_point(alpha = .5) +  
scale_y_log10() +  
xlab('Chromosome 1 [bp]')
```



# Motivation normalization

```
scatter.plot ←  
airway %>%  
assay() %>%  
as_tibble() %>%  
ggplot(aes(x = SRR1039517,  
           y = SRR1039513)) +  
geom_point(alpha = 0.5) +  
geom_abline(slope = 1,  
            color = 'blue') +  
scale_x_log10() +  
scale_y_log10()
```



# Which waterway has more fish?



Photos by Mega Caesaria and Rick Wallace on Unsplash

# RNA-seq data normalization

Potential method, size-factor normalization:

Per library (column) compute the median ratio of gene counts relative to geometric mean per gene (row)

Suggested reading:

Anders S and Huber W. Differential expression analysis for sequence count data. Nature Preceding 2010

RNA-seq analysis workflow tutorial on Bioconductor:

<https://bioconductor.org/packages/release/workflows/vignettes/rnaseqGene/inst/doc/rnaseqGene.html>

# Differential expression analysis with {DESeq2}

```
library(DESeq2)

des ←
  airway ▷
  # convert to DataSet for DESeq
  # analyse expression by cell line and dex treatment
  DESeqDataSet(design = ~ cell + dex) ▷
  # Run DESeq2 normalization and regression models
  DESeq()

# Statistical results
res ← results(des, tidy = TRUE)
```

RNA-seq analysis workflow tutorial on Bioconductor:

<https://bioconductor.org/packages/release/workflows/vignettes/rnaseqGene/inst/doc/rnaseqGene.html>

# Statistical analysis of expression changes

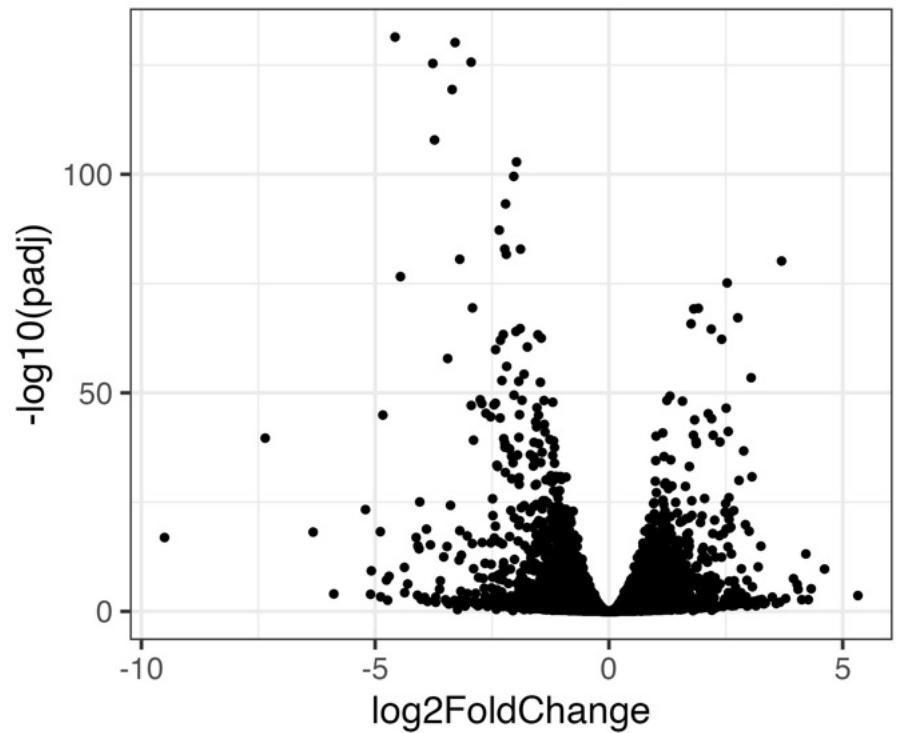
$$\text{Expression Fold Change} = \frac{\text{Expression in condition A}}{\text{Expression in condition B}}$$

A	B	Fold Change	log2
100	100	1.0	0
100	50	2.0	1
100	200	0.5	-1

Testing for statistical significance and false discovery rate (FDR) adjustment

# Analysis results

```
res >  
  ggplot(aes(log2FoldChange, - log10(padj)))  
  geom_point() +  
  theme_bw(18)  
  
ggsave('volcano.jpeg')
```



# Hello academic paper writing phase

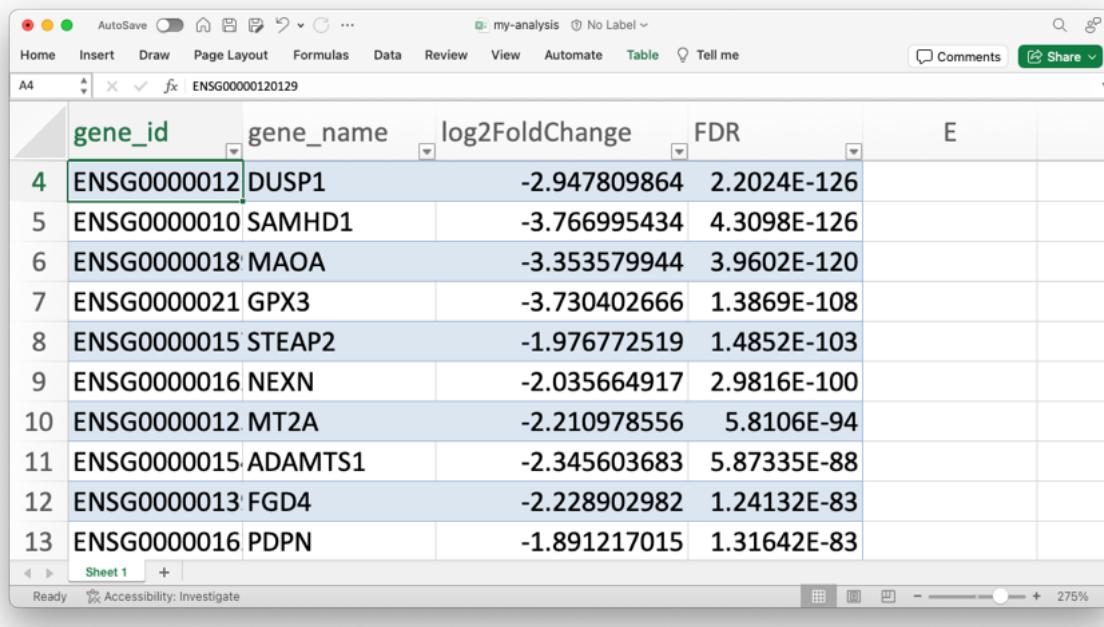
```
my.analysis <-  
  res %>  
  filter(padj <= 0.05, abs(log2FoldChange) >= 1) %>  
  left_join(airway %> rowData() %> as_tibble(), c('row' = 'gene_id')) %>  
  select(gene_id = row, gene_name, log2FoldChange, FDR = padj) %>  
  arrange(FDR)
```

gene_id	gene_name	log2FoldChange	FDR
ENSG00000152583	SPARCL1	-4.57	< 1e-100
ENSG00000165995	CACNB2	-3.29	< 1e-100
ENSG00000120129	DUSP1	-2.94	< 1e-100
ENSG00000101347	SAMHD1	-3.76	< 1e-100
ENSG00000189221	MAOA	-3.35	< 1e-100
ENSG00000211445	GPX3	-3.73	< 1e-100

# Dealing with the office devil {openxlsx}

```
library(openxlsx)
```

```
my.analysis ▷  
write.xlsx('my-analysis.xlsx')
```



A screenshot of Microsoft Excel showing a table titled "my-analysis". The table has columns: gene\_id, gene\_name, log2FoldChange, FDR, and E. The rows contain data for genes DUSP1, SAMHD1, MAOA, GPX3, STEAP2, NEXN, MT2A, ADAMTS1, FGD4, and PDPN. The first row (DUSP1) is selected.

	gene_id	gene_name	log2FoldChange	FDR	E
4	ENSG00000120129	DUSP1	-2.947809864	2.2024E-126	
5	ENSG0000010	SAMHD1	-3.766995434	4.3098E-126	
6	ENSG0000018	MAOA	-3.353579944	3.9602E-120	
7	ENSG0000021	GPX3	-3.730402666	1.3869E-108	
8	ENSG0000015	STEAP2	-1.976772519	1.4852E-103	
9	ENSG0000016	NEXN	-2.035664917	2.9816E-100	
10	ENSG0000012	MT2A	-2.210978556	5.8106E-94	
11	ENSG0000015	ADAMTS1	-2.345603683	5.87335E-88	
12	ENSG0000013	FGD4	-2.228902982	1.24132E-83	
13	ENSG0000016	PDPN	-1.891217015	1.31642E-83	



Happy customer!

Photo by Stephen Baker on Unsplash



```
rule my_analysis:  
    input:  
        script = 'analysis.R',  
        data = 'gene-expression-matrix.tsv'  
    output:  
        'volcano.jpeg',  
        'my-analysis.xlsx'  
    shell:  
        "RScript {input.script}"
```

- Pro tip: Define script as input (triggers rerun on changes)
- Explicitly list data input, such that script gets executed at the right time
- Increase reproducibility by declaring software environment with conda

Snakemake Handbook:

<https://snakemake.readthedocs.io/en/stable/snakefiles/deployment.html#integrated-package-management>



There is no "one size fits all"  
expression analysis workflow

Photo by Julian Hochgesang on Unsplash

# Presentation takeaways

- Define "*transcriptomic*" sequencing and its relevance
- Recognize structure of sequencing data
- Outline a simple analysis pipeline with bash
- Identify concept and advantages of a Snakemake workflow  
(How I avoid using bash as much as possible)
- Connect a workflow with R analysis scripts
- Introduction to software management across platforms
- Fun with  packages and R tweaks



Thank you for joining!

Photo by Peter Lloyd on Unsplash



Supervisors: SE Seemann, J Gorodkin  
Independent Research Fund Denmark

Photo by Peter Lloyd on Unsplash