RNA-seq data analysis

A very short course

By Li Shen

http://labs.neuroscience.mssm.edu/project/shen-lab/

Department of Neuroscience and Artificial Intelligence and Human Health

What will be covered in this course?

- A very brief introduction to RNA-seq analysis.
- Two modules: introduction to various tools & an interactive session.
- Topics covered: data download from GEO, QC, alignment, abundance estimation, differential gene expression analysis.

What will NOT be covered in this course?

- Single-cell RNA-seq
- Network analysis such as WGCNA
- Niche applications: small RNAs, long-read sequencing, 5’ and 3’ ends analysis, alternative splicing analysis, etc.

The central dogma and gene expression

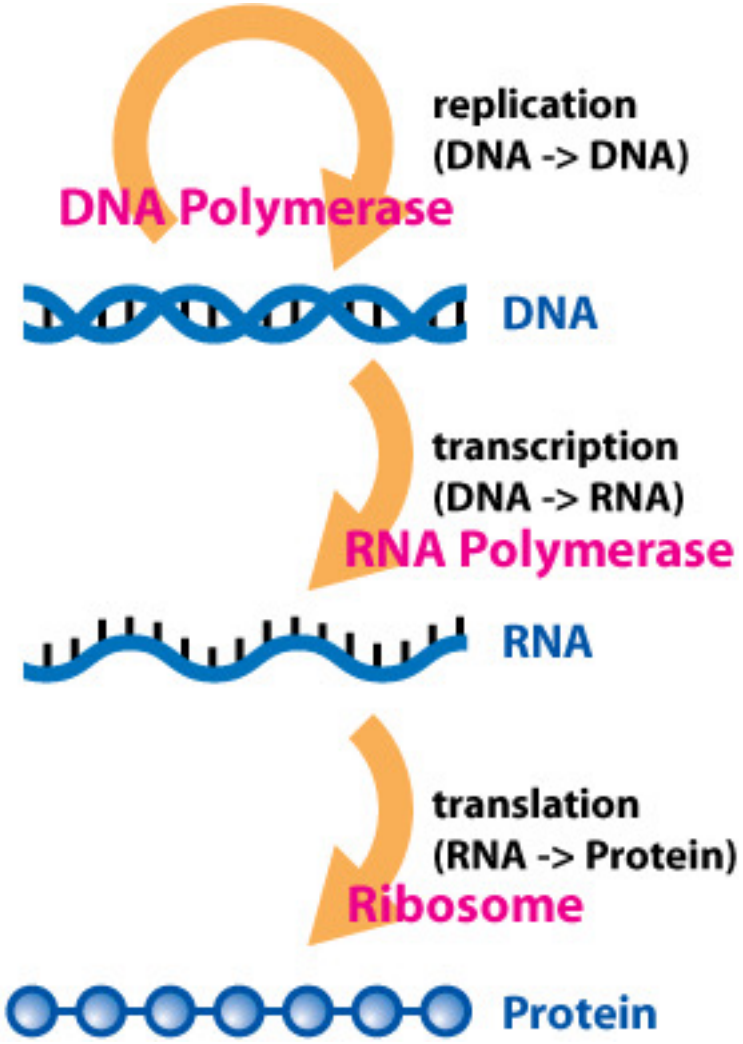


Fig. credit: Daniel Horspool

Gene expression: control the flow of information from genomic DNA to functional protein products.

RNA-seq

Pros	Cons
Highly accurate: compared with PCR and microarrays	It's a snapshot of the system
High sensitivity	Doesn't always correlate with protein expression
Powerful: measures various RNA products unbiasedly; sequence first, then analyze	Biases can be introduced during various steps
	Starting material requirement can be high

The lifecycle of an RNA-seq experiment

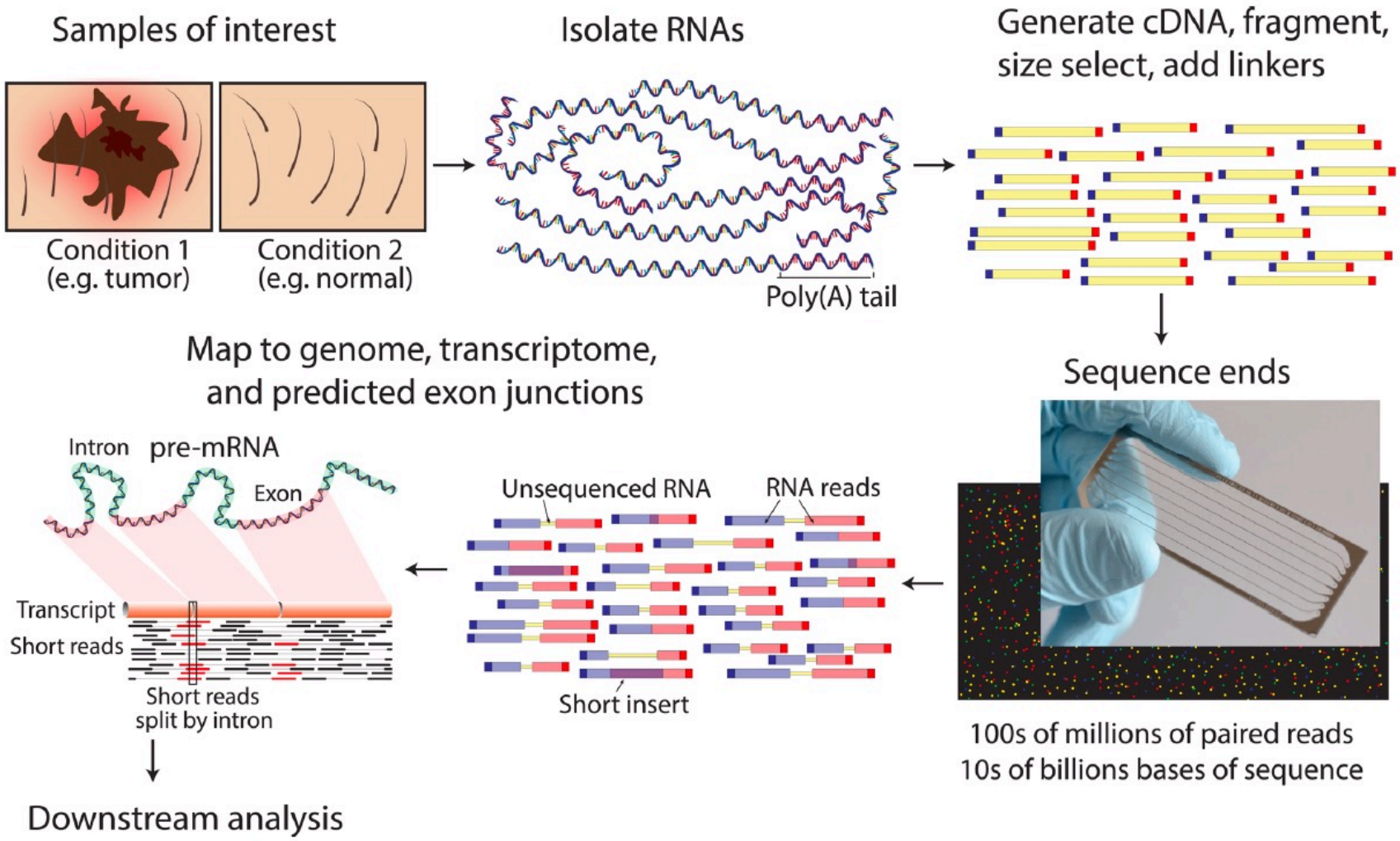


Fig. 2 from Griffith M, et al. (2015) PLoS Comput Biol 11(8): e1004393.

RNA enrichment methods

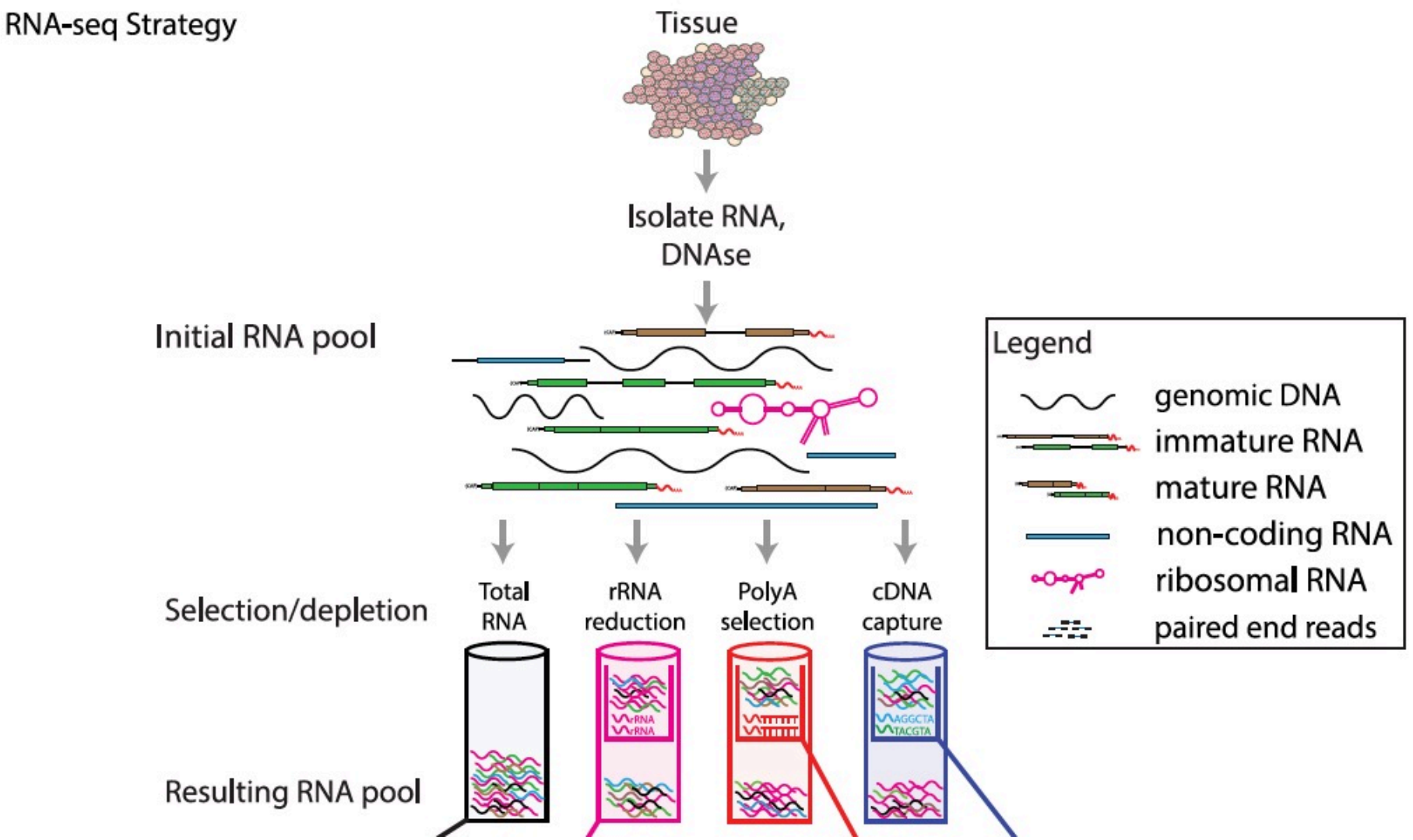


Fig. 4 from Griffith M, et al. (2015) PLoS Comput Biol 11(8): e1004393.

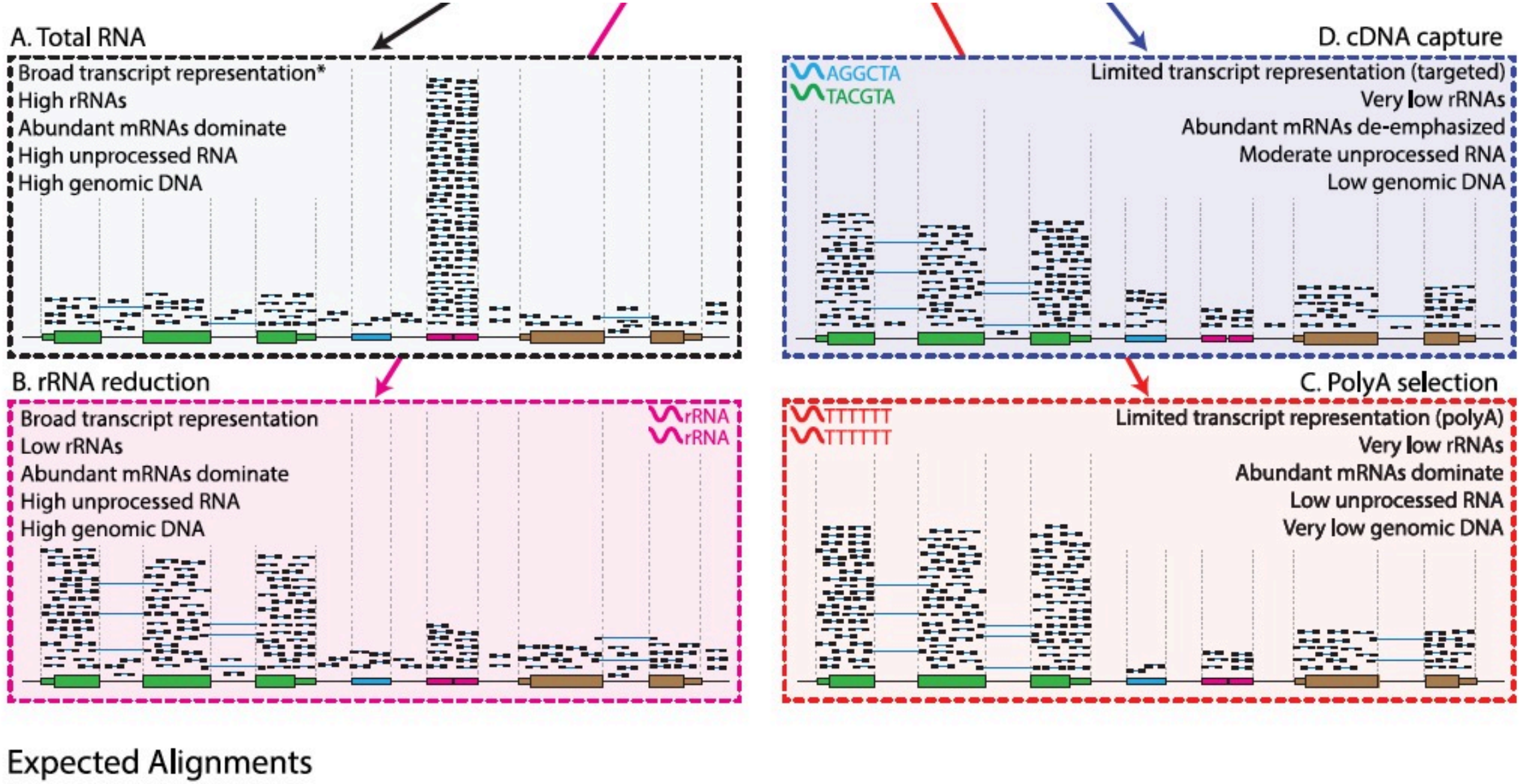
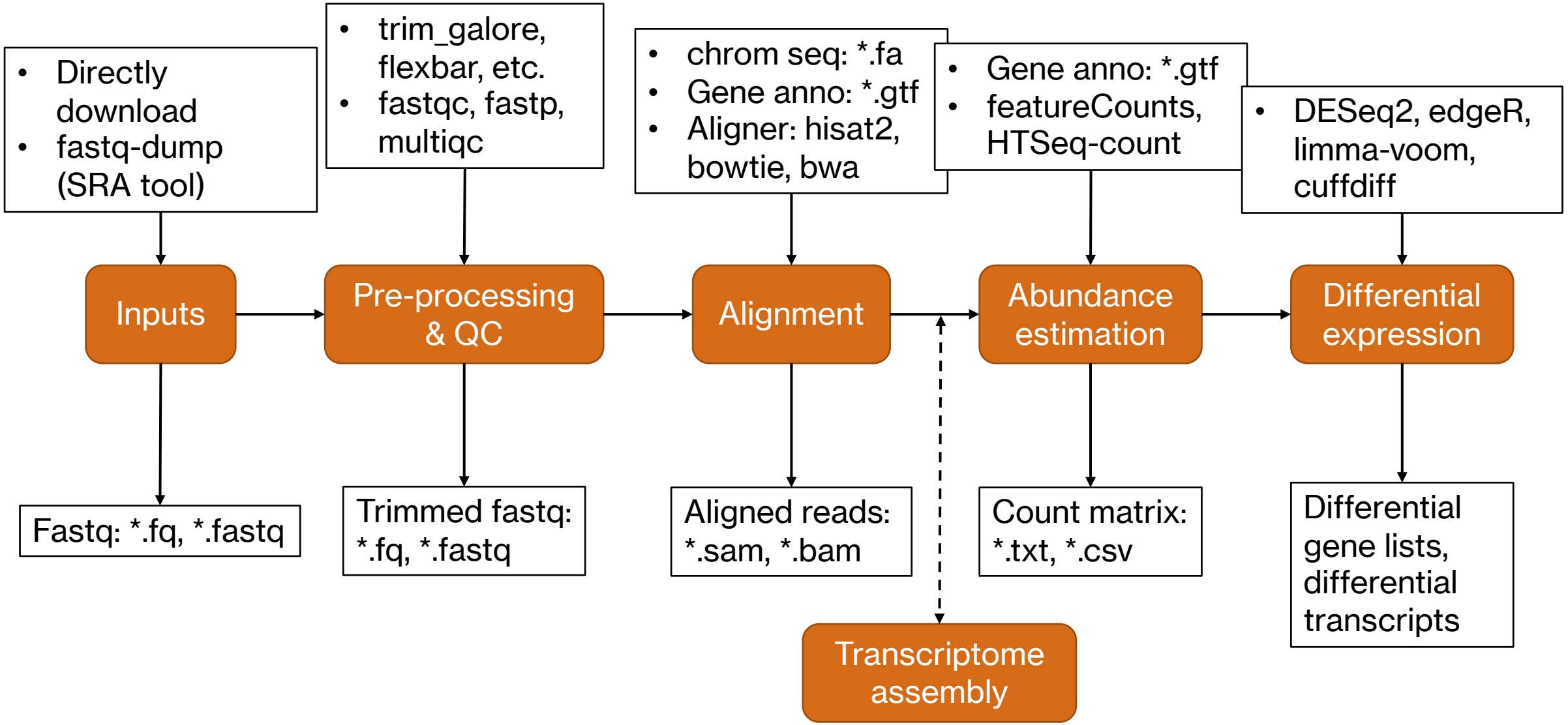
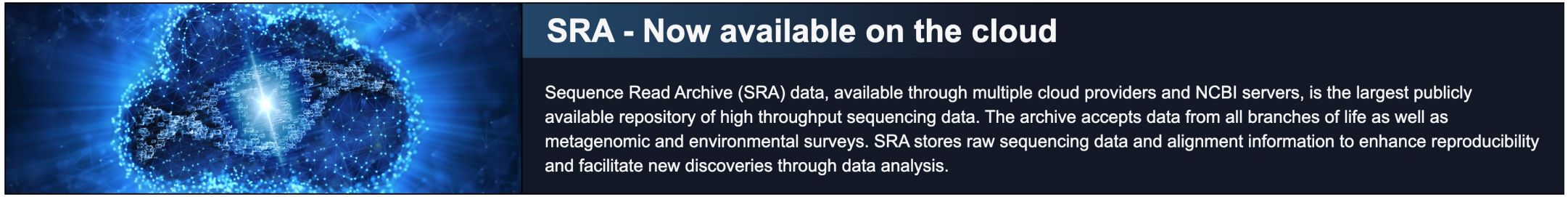


Fig. 4 from Griffith M, et al. (2015) PLoS Comput Biol 11(8): e1004393.


• Directly download
• fastq-dump (SRA tool)
• trim_galore, flexbar, etc.
• fastqc, fastp, multiqc
• chrom seq: *.fa
• Gene anno: *.gtf
• Aligner: hisat2, bowtie, bwa
• Gene anno: *.gtf
• featureCounts, HTSeq-count
• DESeq2, edgeR, limma-voom, cuffdiff
Inputs
Pre-processing & QC
Alignment
Abundance estimation
Differential expression
Fastq: *.fq, *.fastq
Trimmed fastq: *.fq, *.fastq
Aligned reads: *.sam, *.bam
Count matrix: *.txt, *.csv
Differential gene lists, differential transcripts
Transcriptome assembly

Where do I get my sequence data?

- If you are doing your own experiments: FTP, hard-drive, USB disk
- If you are using a public dataset:



What is SRA?

- Sequence Read Archive (SRA) is the world's largest publicly available repository of high throughput sequencing data.
- You can submit any biological sequences to SRA!
- Also available via Google cloud and AWS.

GEO
Gene Expression Omnibus

- GEO is a frontend for SRA with a focus on functional genomics.
- Accept data such as RNA-seq, ChIP-seq and microarray.
- The aim is to make data depository and retrieval as easy as possible.

What is a fastq file?

- Commonly used file format for NGS data.
- A text file that contains both sequences and the corresponding quality information.

An example read:

@SRR8956207.1 HWI-D00381:363:C8DHEANXX:3:1101:1628:1998 length=125 ← Read ID
NGTCAGTGGCCTCAAATGTCTGCATCACACCTATCGTCGGGGACTGCGTGTG
GACAAGGCCATCTTCATGGTGGGAAGCTACGGCCCCAGAGCCTAGGAAGAT ← Nucleotides
CGGAAGAGCACACGTCTGAACT
+SRR8956207.1 HWI-D00381:363:C8DHEANXX:3:1101:1628:1998 length=125 ← Same ID
#<=ABFGGGGEGGGGGGFFCGGGGGGGGGGGGGGGGGGGGGEGGGGGG
GGGGGGDGGGGGGGGGGGGGGGGGFGGGGGGGGGGGGGGBGGGGGGGG ← Quality values
GG=GGGGGEGCGGGGGGGGGG.->@@G#

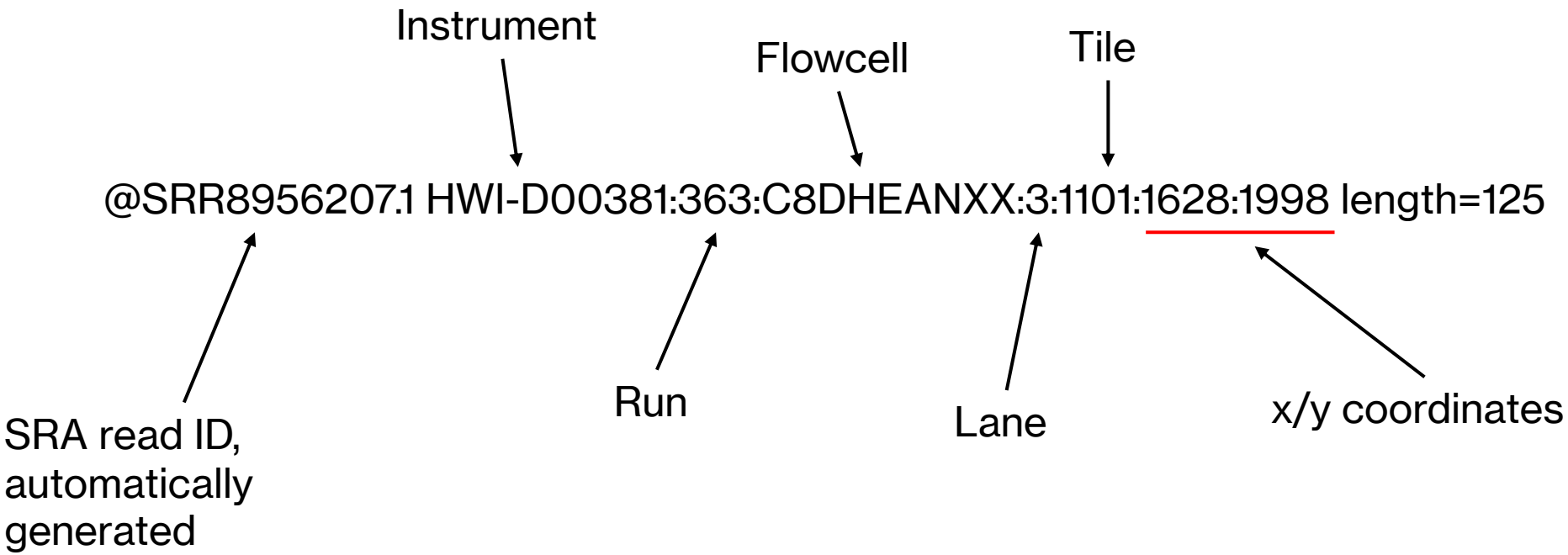
How do you download fastq files from SRA? (use fastq-dump)

Basic usage: `fastq-dump --split-3 SRR8956208`

Split the mate-pair into two

SRA accession

The read ID explained



The quality values explained

#<=ABFGGGGEGGGGGGFFCGGGGGGGGGGGGGGG
GGGGGGGGGEGGGGGGGGGGGGGDGGGGGGGGGGG
GGGGGGFGGGGGGGGGGGGGGBGGGGGGGGGG=G
GGGGEGCGGGGGGGGGGG.->@@G#

lookup → ASCII table → convert → Phred score

What is a Phred score?

$$\text{Phred score} = -\log_{10}(error\ rate) \times 10$$

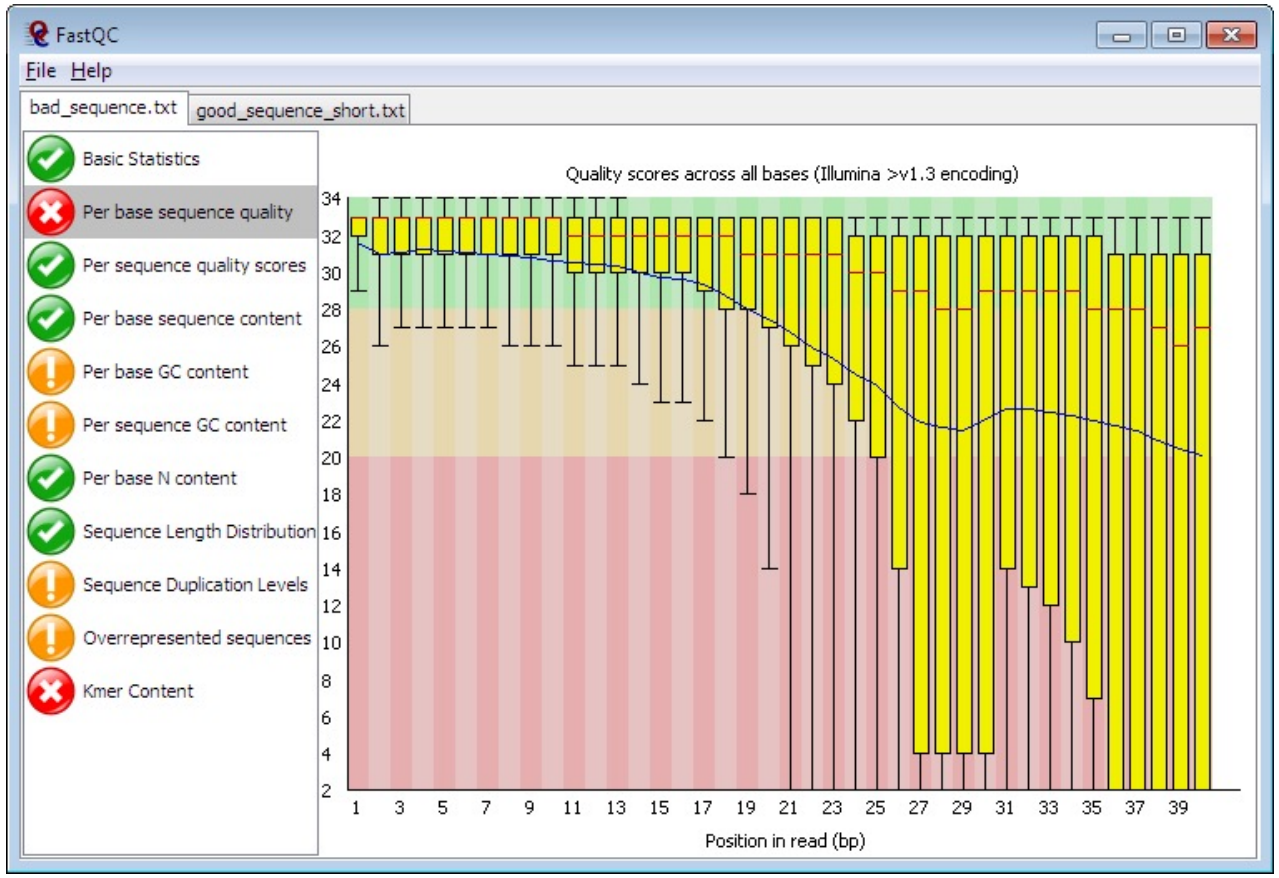
Phred score	Error rate
10	0.1
20	0.01
30	0.001
40	0.0001

Sequence quality check

FastQC aims to provide a simple way to do some quality control checks on raw sequence data. It gives you a quick impression of whether your data has any problems.

https://www.bioinformatics.babraham.ac.uk/projects/fastqc/

Basic usage: fastqc X_1.fastq X_2.fastq -o /path/output/fastqc/



Adapter and quality trimming

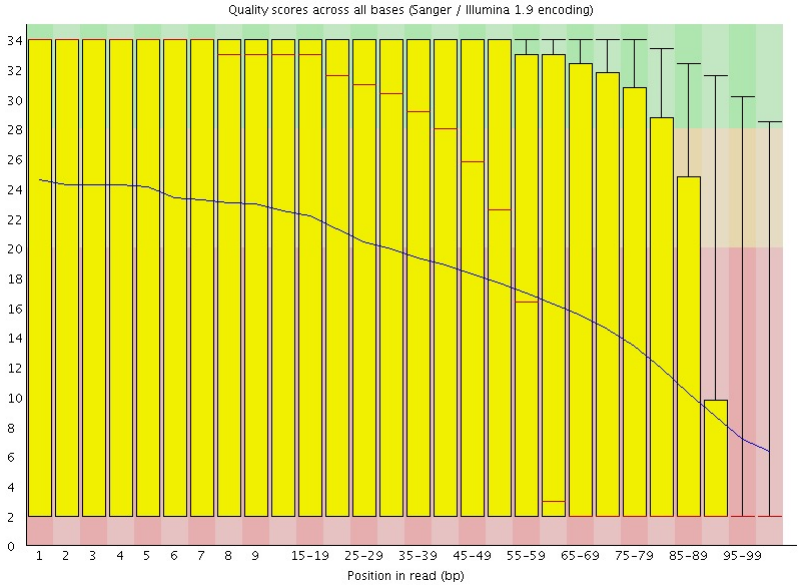
- Adapter trimming: adapters are attached to cDNA in library construction.
- Quality trimming: both ends of a read typically suffer quality drop.

Trim Galore! – "A wrapper tool around Cutadapt and FastQC to consistently apply quality and adapter trimming to FastQ files, with some extra functionality for MspI-digested RRBS-type (Reduced Representation Bisufite-Seq) libraries."

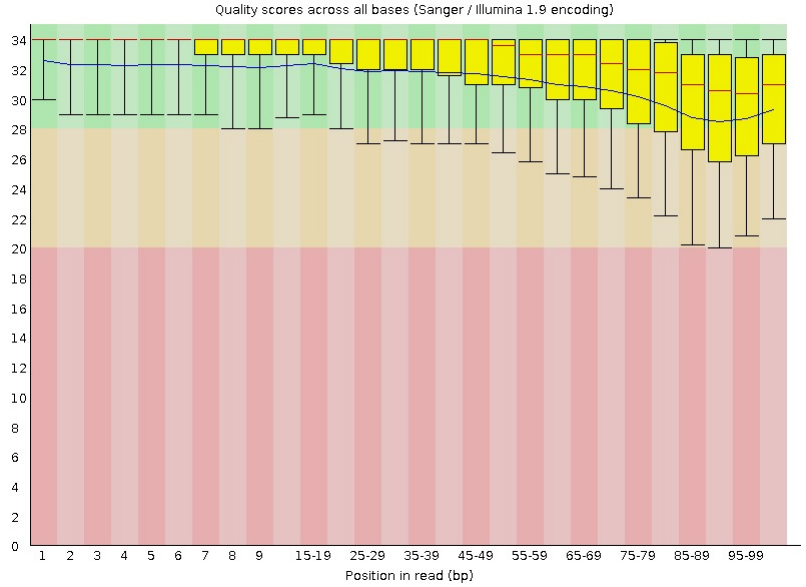
Basic usage: trim_galore --gzip --paired XXX_1.fastq XXX_2.fastq -o /path/output/trim_fastq

Zip the output files Read 1 Read 2

Before



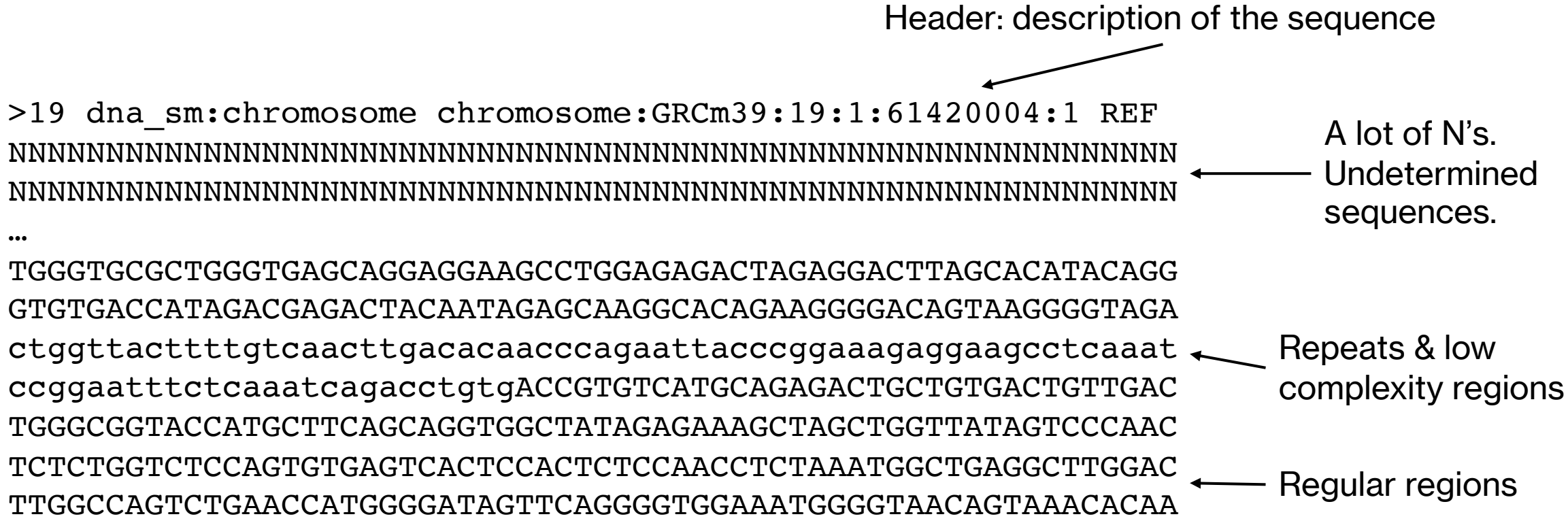
After



https://github.com/FelixKrueger/TrimGalore/blob/master/Docs/Trim_Galore_User_Guide.md

Before alignment: build the index for the reference genome

Chromosomal sequences
fasta: a format for nucleic acid or amino acid sequences
Fasta = fastq – q

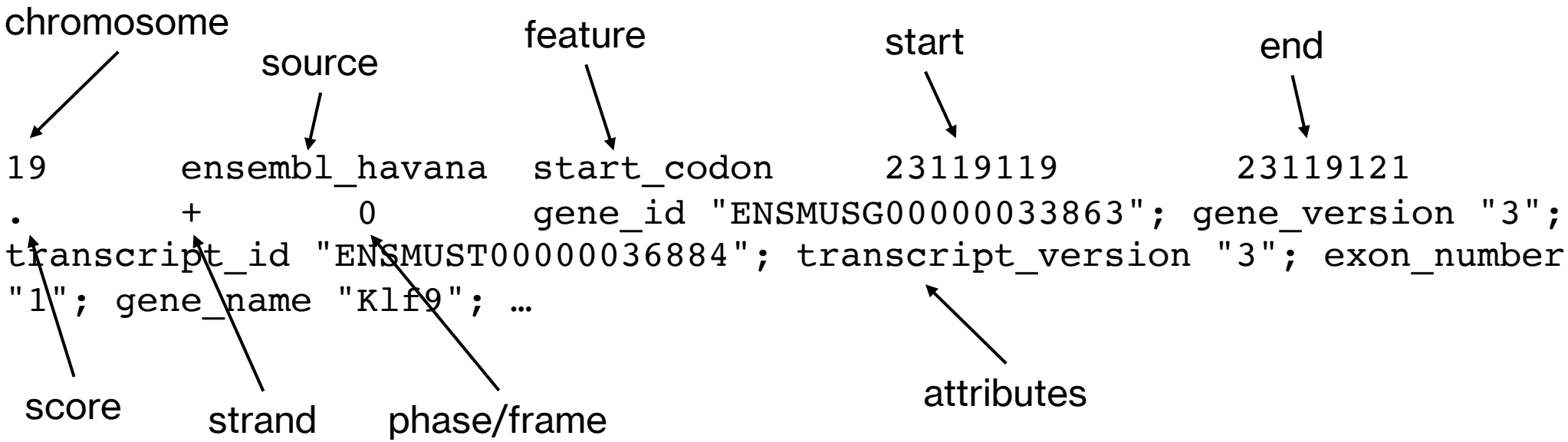


GTF: a format for the transcriptome

GTF – Gene Transfer Format, tab-separate files with 9 fields

https://en.wikipedia.org/wiki/General_feature_format

An example GTF record:



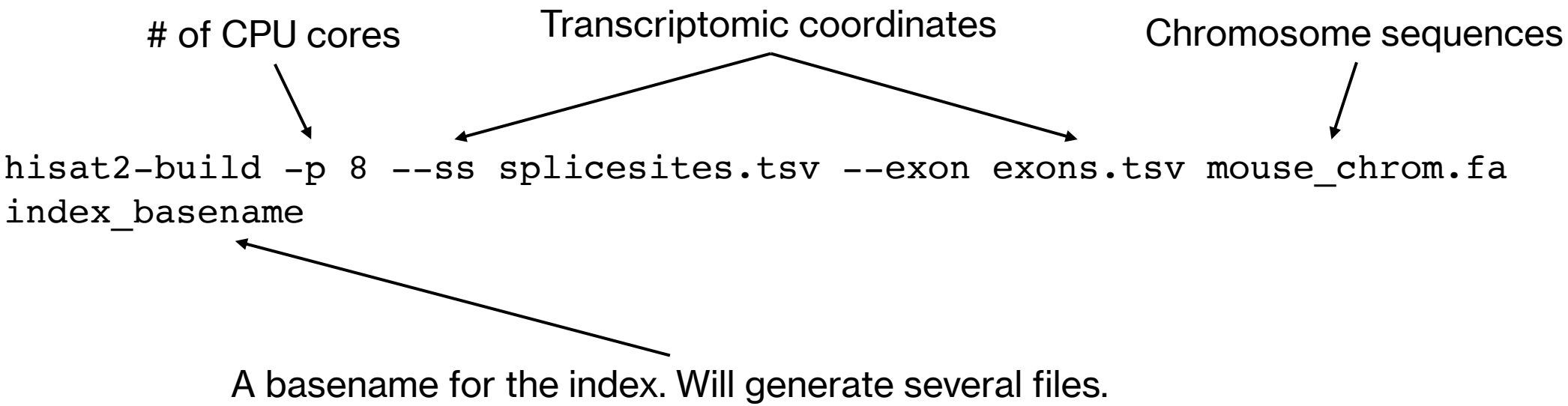
Transcript = 5'UTR + start codon + exons + end codon + 3'UTR
Many records per transcript per gene

Build a transcriptome index using HISAT2

An example

Extract coordinates for splice sites and exons:

```
hisat2_extract_splice_sites.py mouse_chrom.gtf > splicesites.tsv
hisat2_extract_exons.py mouse_chrom.gtf > exons.tsv
```

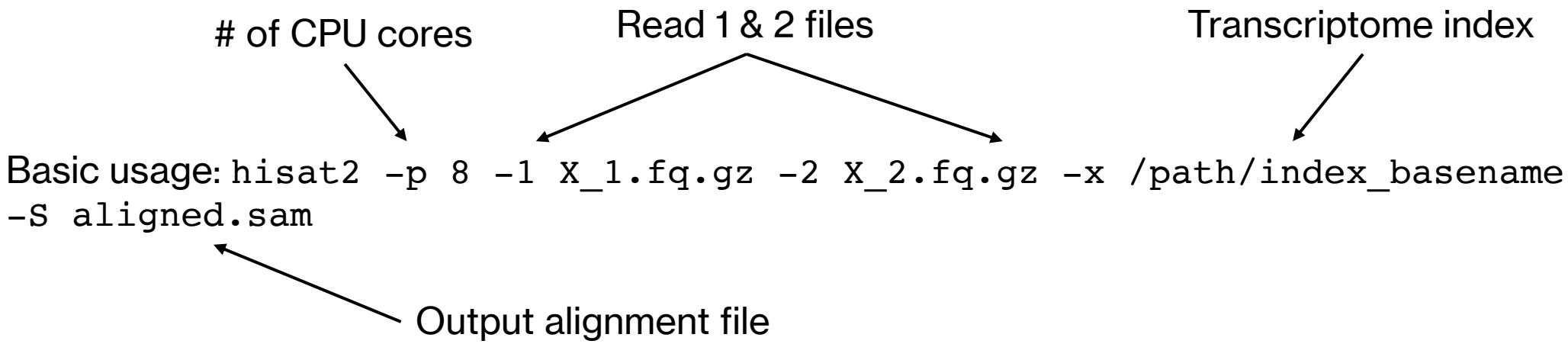


Short read alignment

Summary of popular aligners

Aligner	Accuracy	Time	Memory	Splice aware?
Bowtie, BWA	Higher	Long	Low	No
STAR	High	Short	High	Yes
HISAT/HISAT2	High	Short	Medium	Yes

An example of HISAT2



What is the SAM/BAM format?

- SAM is tab-delimited text format for read alignment.
- BAM is compressed binary version of SAM (~3-4x smaller).
- BAM can be indexed (.bai) for fast retrieval of alignments.
- Full specification: http://samtools.sourceforge.net/SAM1.pdf

The SAM/BAM file has two parts – header and alignment.

Here is an example of the header:

```
@HD     VN:1.0  SO:coordinate
@SQ     SN:19   LN:61420004
@PG     ID:hisat2       PN:hisat2       VN:2.2.1        CL:"/hpc/packages/min
erva-centos7/hisat2/2.2.1/src/hisat2-2.2.1/hisat2-align-s --wrapper basic-0 -
p 8 -x ../genome/mouse/Mus_musculus.GRCm39.105.chr19 -S bam/SRR8956207.sam --
read-lengths
125,124,123,122,121,120,119,118,116,117,115,113,110,114,112,109,111,108,106,9
9,105,107,103,102,104,101,100,94,95,98,92,97,91,90,87,84,88,86,80,78,75,48,82
,79,51,93,89,85,81,73,69,68,64,57,54,45,36,96,83,77,72,70,67,66,65,63,56,55,4
9,44,40,38,32,31,25,22 -1 /tmp/445923.inpipe1 -2 /tmp/445923.inpipe2"
@PG     ID:samtools     PN:samtools     PP:hisat2       VN:1.13 CL:samtools
sort -@ 8 -o SRR8956207.bam SRR8956207.sam
```

Example alignments (2 read-pairs)

```
SRR8956207.13    83      19      9962162 60      125M    =       9961992 -
200    GACGTATTATCTGAGTGAACAGGTGAAATCCATTAAAGAACTGGGTGACCACGTGACCAACTTACGCAAGATGGGTGCCCC
TGAAGCTGGCATGGCAGAATATCTCTTTGACAAGCACACCCTGN   BGGC/F@GF0GF@>F0F0G@0CFGFFE80>FF;CEEGGFGG
BDCFF:@GGF>GF;C@>@F>GF<CBDE11GGF@DGDBF<1@GEEGF>CF1@GGEGGGC1FFBDCEGGGEGFFDE/EE?;0:<:#   A
S:i:-1 ZS:i:-1 XN:i:0  XM:i:1  XO:i:0  XG:i:0  NM:i:1  MD:Z:124G0      YS:i:-3 YT:Z:CP
NH:i:1
SRR8956207.13    163     19      9961992 60      58M95N48M       =       9962162
200    TGTGAATCAGTCACTACTGGAACTGCACAAACTGGCTACAGACAAGAATGATCCCCACTTATGTGACTTCATTGAGACGT
ATTATCTGAGTGAACAGGTGAAATCC      :3:A0FGCG>FGG1BFG@1C@>>FG@FGEEG0::DG>1?1=<F1=FDCDGEGEGGB
GGGEEGGGG>G1=01EDF00E@@FGBCGEC0?/;B?/;/<F//:BFG/DC      AS:i:-3
XN:i:0  XM:i:1  XO:i:0  XG:i:0  NM:i:1  MD:Z:39T66      YS:i:-1 YT:Z:CP XS:A:+  NH:i:1
SRR8956207.52    99      19      43480110        60      125M    =       43480172
187    CTCCTCCACTTCTGCCTAGCAGGCATTTTGTGCCTCACAGAGCCCAAGGTCTCCTCCCAGGCACAGAGGTAACCTACTCG
CTGTCCTCCATAGGCAGCCGCTCGGATGGTCTGGGAAGCCCAGGC   AA300=FGG@DFD@C1EFGGGGGDBGEGGGEB1BDC:F1E
:FGGGE>D0FF:F1EB1FFGED0E@1F10=00=CF0F@@//0=F0;0:@;0E0:CFG/8/CCDA.CC.@66@0-A0.,9-
7;BBB   AS:i:-6 ZS:i:-
15        XN:i:0  XM:i:2  XO:i:0  XG:i:0  NM:i:2  MD:Z:79A33A11   YS:i:-6 YT:Z:CP NH:i:1
SRR8956207.52    147     19      43480172        60      125M    =       43480110
-
187    ACAGAGCTAACCTACTCACTGTCCTCCATAGGCAGCCGCTCGGATGGTCTGGGAAGCCCAGGCGAGGTTGTGTACCTGAGG
ATGGAGGAGATGGCCTTCCCCCAAGAAGAGATGCCCAACTTTGA   GB.7.7-.@.GB@75.=?3-
55BGEBCGGF5,,,,@:4/GGF@CF0B0F=000GF0DEGGGC=0:/CGFE0FGGF=0F?01E::@CGGF>G><>DF=/GGGGGGF1GC
EGFBBF>1;1>@00<>A   AS:i:-6 ZS:i:-10        XN:i:0  XM:i:2  XO:i:0  XG:i:0
```

SAM/BAM alignment specification

Col	Field	Type	Regexp/Range	Brief description
1	QNAME	String	`[!-?A-~]{1,255}`	Query template NAME
2	FLAG	Int	$[0,2^{16}-1]$	bitwise FLAG
3	RNAME	String	`\*|[!-()+-<>-~][!-~]*`	Reference sequence NAME
4	POS	Int	$[0,2^{29}-1]$	1-based leftmost mapping POSition
5	MAPQ	Int	$[0,2^{8}-1]$	MAPping Quality
6	CIGAR	String	`\*|([0-9]+[MIDNSHPX=])+`	CIGAR string
7	RNEXT	String	`\*|=|[!-()+-<>-~][!-~]*`	Ref. name of the mate/next segment
8	PNEXT	Int	$[0,2^{29}-1]$	Position of the mate/next segment
9	TLEN	Int	$[-2^{29}+1,2^{29}-1]$	observed Template LENgth
10	SEQ	String	`\*|[A-Za-z=.]+`	segment SEQuence
11	QUAL	String	`[!-~]+`	ASCII of Phred-scaled base QUALity+33

1	QNAME	SRR895620713 (read name)
2	FLAG	163 (will explain next)
3	RNAME	19 (ref. name, i.e. chromosome name)
4	POS	9961992 (chromosomal position)
5	MAPQ	60 (Phred-scale score)
6	CIGAR	58M95N48M (will explain next)
7	RNEXT	= (mate pair name, “=“ means the same)
8	PNEXT	9962162 (mate pair position)
9	TLEN	200 (insert size)
10	SEQ	TGTGAATCAGTCACTACTGGAACTGCACAAACTGGCTAC...
11	QUAL	:3:A0FGCG>FGG1BFG@1C@>>FG@FGEEG0::DG>1?1=<F1...

The alignment FLAG explained

- The FLAG is a 12-bit binary string describing the alignment.
- E.g. if both bit 1 & 2 are set, `0b000000000011 = 3`

Bit		Description
1	0x1	template having multiple segments in sequencing
2	0x2	each segment properly aligned according to the aligner
4	0x4	segment unmapped
8	0x8	next segment in the template unmapped
16	0x10	SEQ being reverse complemented
32	0x20	SEQ of the next segment in the template being reverse complemented
64	0x40	the first segment in the template
128	0x80	the last segment in the template
256	0x100	secondary alignment
512	0x200	not passing filters, such as platform/vendor quality controls
1024	0x400	PCR or optical duplicate
2048	0x800	supplementary alignment

A recipe to find the paired-end reads that are properly aligned:

I want the good stuffs — Filter out the bad stuffs

```
samtools view -f 3 -F 1804 my_aligned.bam
```

Bit 1 + 2 — Bit 4 + 8 + 256 + 512 + 1024

Identify the unmapped reads (but are not low-quality or duplicates):

```
samtools view -f 4 -F 1536 my_aligned.bam > unmapped.sam
```

Bit 512 + 1024

→ To use for BLAST, etc.

The CIGAR string explained

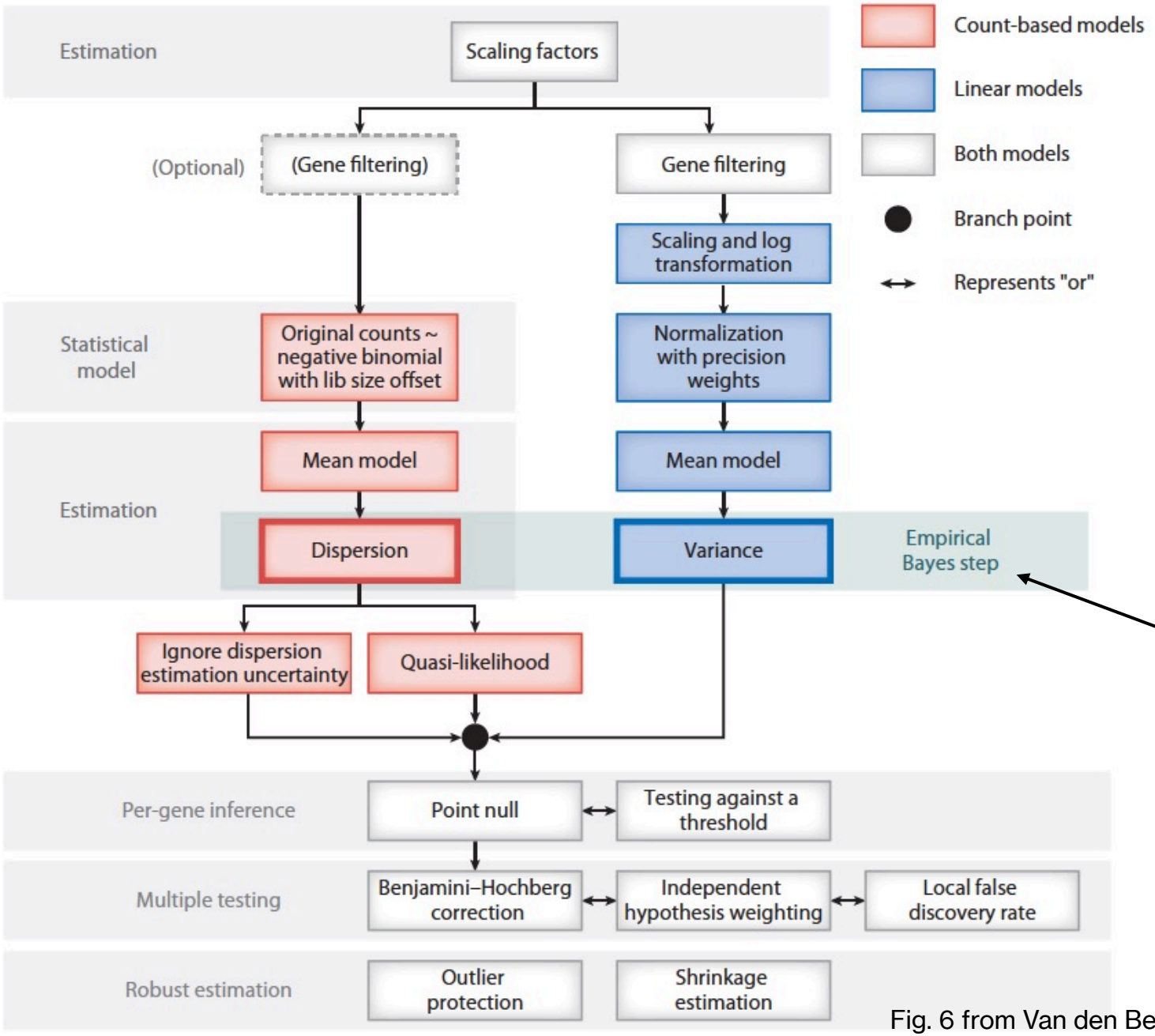
- CIGAR=**C**ompact **I**diosyncratic **G**apped **A**lignment **R**eport
- The CIGAR string is a sequence of base sizes and associated operations describing the alignment.

Op	BAM	Description
M	0	alignment match (can be a sequence match or mismatch)
I	1	insertion to the reference
D	2	deletion from the reference
N	3	skipped region from the reference
S	4	soft clipping (clipped sequences present in SEQ)
H	5	hard clipping (clipped sequences NOT present in SEQ)
P	6	padding (silent deletion from padded reference)
=	7	sequence match
X	8	sequence mismatch

The CIGAR from the previous example: 58M95N48M

- 58 bases of match to the first exon, 95 bases skipped (an intron), and 48 bases of match to the second exon.

Differential gene expression (DGE) analysis for RNA-seq



Two "schools of thoughts":

1. Count-based models
2. Linear-model based methods

Dispersion/variance estimation using EB due to small sample size

Fig. 6 from Van den Berge, K. *et al. Annual Review of Biomedical Data Science* **2**, 139–173 (2019).

Shot noise: read count variation from **technical** replicates (Poisson)
Poisson assumes: var = mean

Higher variance due to **biological** replicates

Negative Binomial (NB) distribution for RNA-seq read counts

$$Y_{fi} \sim \mathrm{NB}(\mu_{fi},\ \varphi_f),$$

$$\mathrm{Var}(Y_{fi}) = \mu_{fi} + \varphi_f \mu_{fi}^2,$$

Dispersion

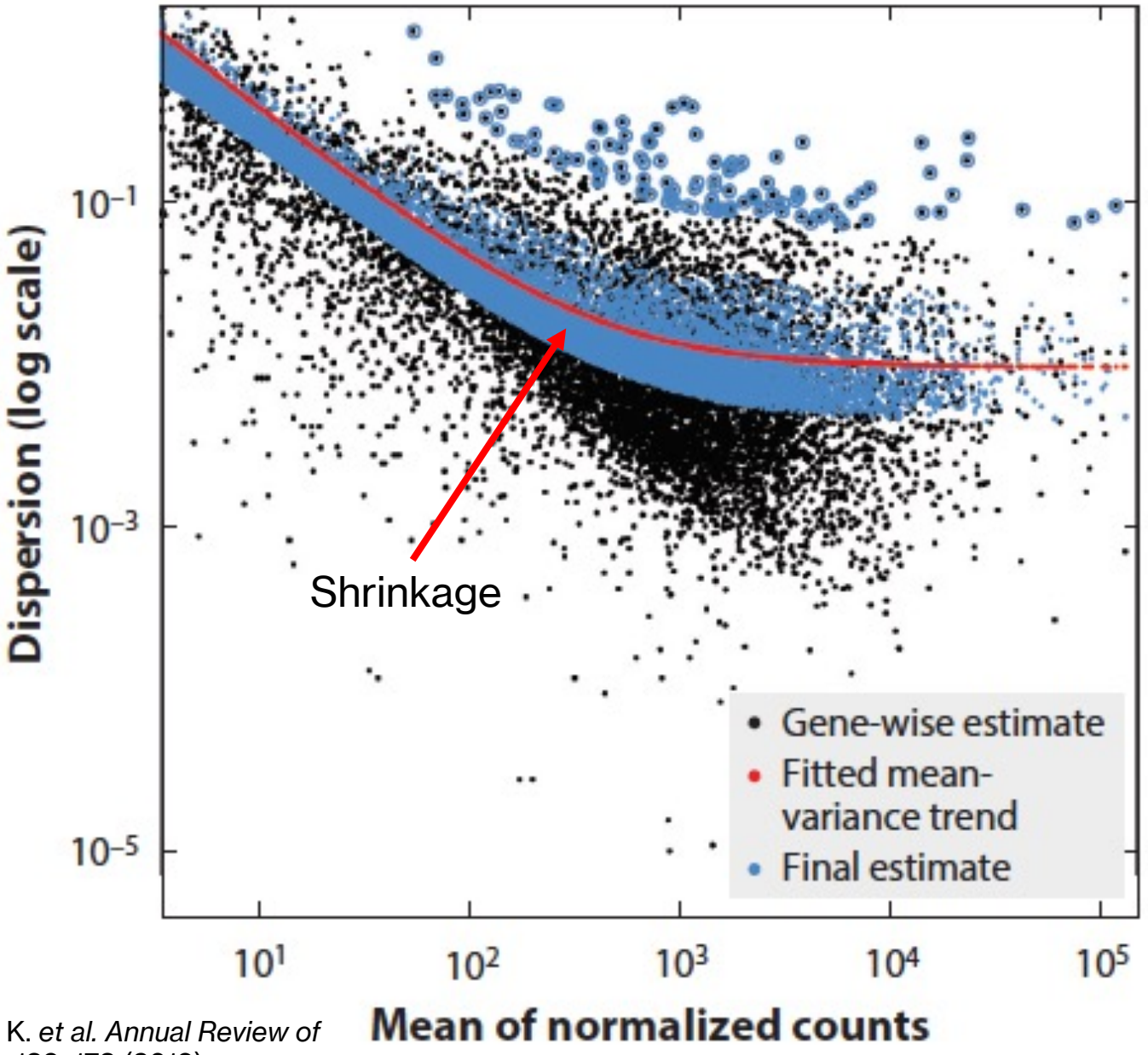
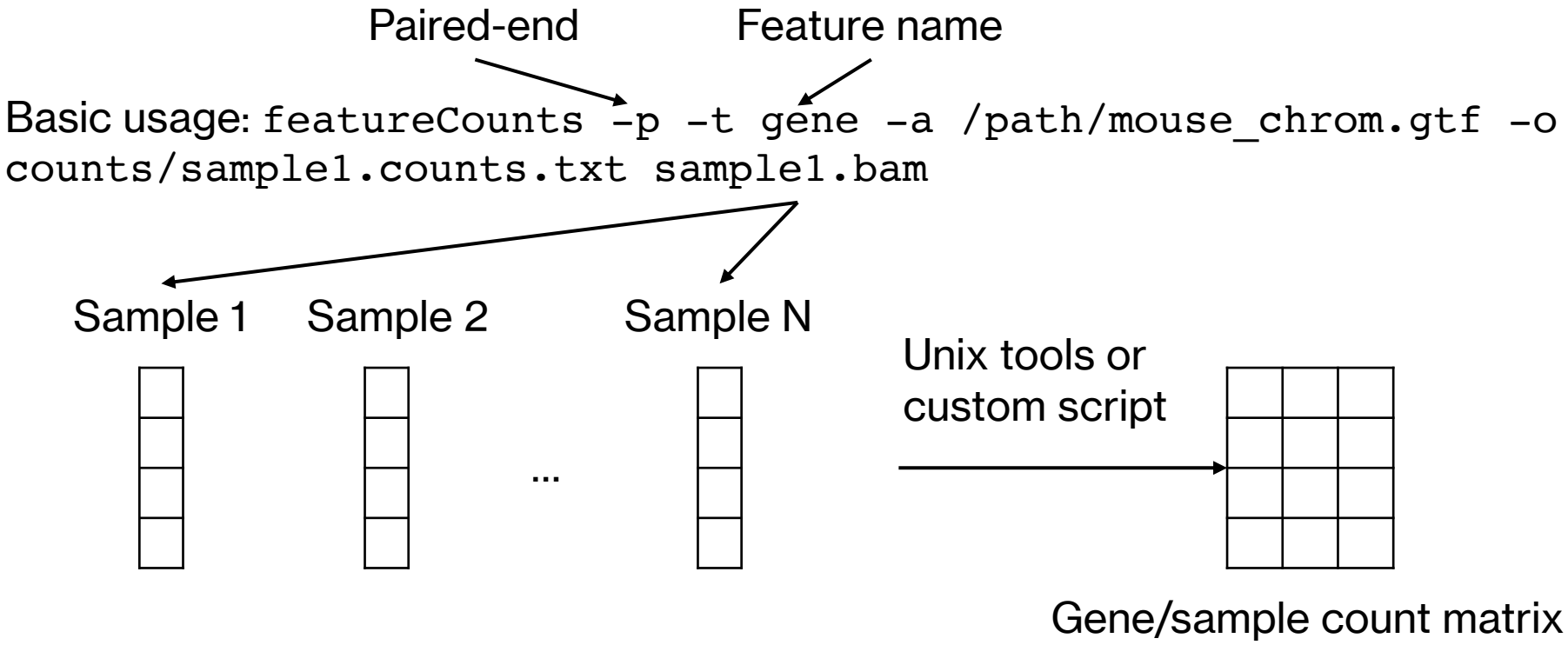


Fig. 7 from Van den Berge, K. *et al. Annual Review of Biomedical Data Science* **2**, 139–173 (2019).

Gene abundance estimation

featureCounts: a ultrafast and accurate read summarization program
http://subread.sourceforge.net/featureCounts.html

- featureCounts is very fast: ~10x faster than HTSeq-Count (another popular tool).
- It is available as a standalone program or R package.
- It can count both RNA-seq and DNA-seq reads on different genomic features.



Differential gene expression analysis with DESeq2

Love, M.I., Huber, W., Anders, S. (2014) Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biology*, **15**:550.

- One of the most popular DGE tools (alternative: edgeR).
- Ranks top position in various benchmarks.

Basic usage of DESeq2:

```
dds <- DESeqDataSetFromMatrix(
       countData = cts,
       colData = coldata,
       design= ~ batch + condition)
```

Count matrix

Sample meta information

How do you want to fit the model?

```
dds <- DESeq(dds)
```

1. Estimate size factors
2. Estimate gene-wise dispersions
3. Mean-dispersion relationship
4. Estimate final dispersions
5. Model fitting and testing

```
resultsNames(dds) # lists the coefficients
res <- results(dds, name="condition_trt_vs_untrt")
```

Extract DGE list

Case study: the impact of early-life stress in adulthood

Peña, C. J. *et al.* Early life stress confers lifelong stress susceptibility in mice via ventral tegmental area OTX2. *Science* **356**, 1185–1188 (2017).

Goal: to study the impact of early life stress on the transcriptome of mouse brain in stress susceptibility in adulthood.

Factor 1 (EarlyLife)	2x2 design	Factor 2 (Stress)
Standard		Control
Early life stress (ELS)		Sub-threshold Variable Stress (SVS)

cts

	SRR8956207	SRR8956208	SRR8956221
ENSMUSG00000001750	8	9	12
ENSMUSG00000003053	1	2	1
ENSMUSG00000003228	193	186	214
ENSMUSG00000003555	0	0	0
ENSMUSG00000003559	10	11	20
ENSMUSG00000003680	9	6	8

coldat

	Run	EarlyLife	Stress
1	SRR8956207	ELS	Control
2	SRR8956208	ELS	SVS
3	SRR8956221	Standard	Control
4	SRR8956222	Standard	SVS
5	SRR8956229	Standard	Control
6	SRR8956230	Standard	SVS

```
dds <- DESeqDataSetFromMatrix(
       countData = cts,
       colData = coldata,
       design= ~ EarlyLife + Stress + EarlyLife:Stress)
```

Interaction

Under the hood of model fitting

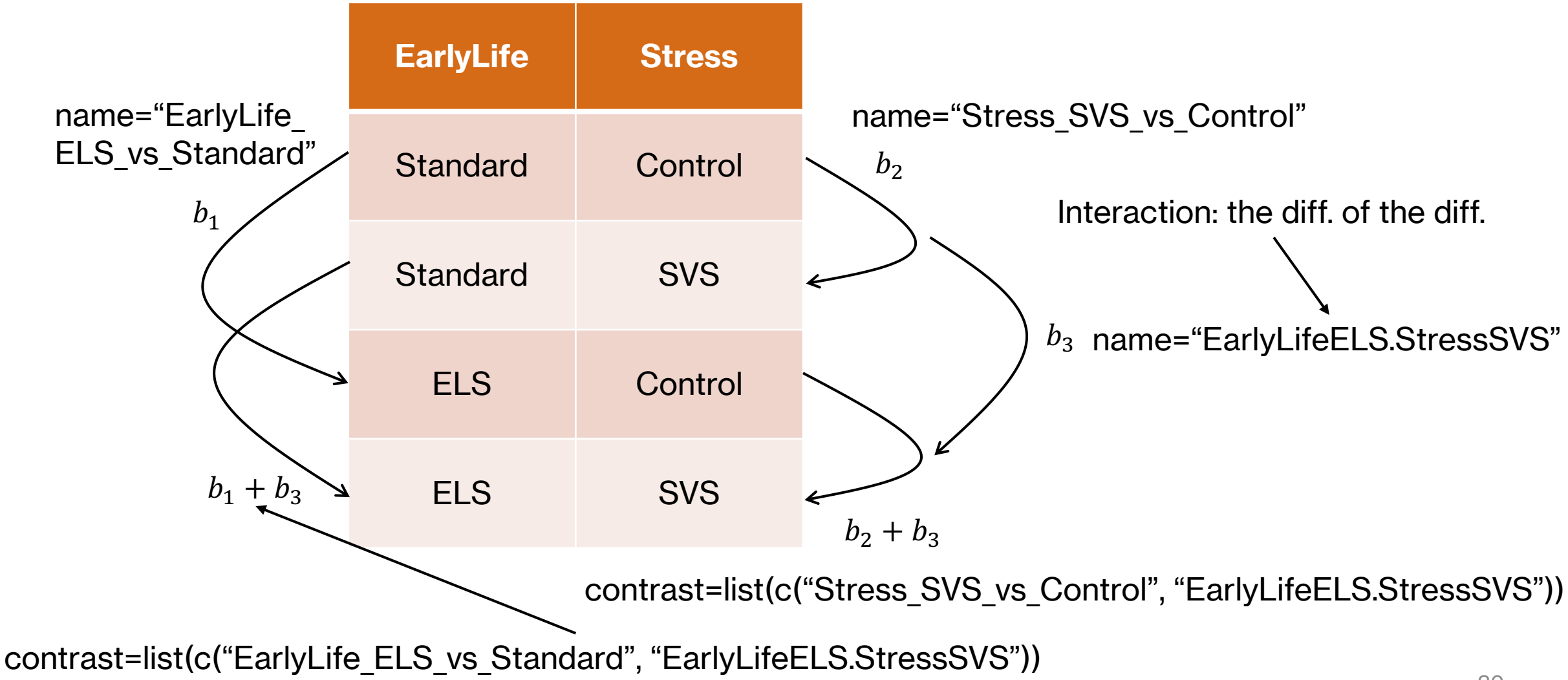
```
Formula: ~ EarlyLife + Stress + EarlyLife:Stress
```

Linear model: $\log(\text{exp}) = b_0 + b_1 \cdot \text{EarlyLife} + b_2 \cdot \text{Stress} + b_3 \cdot \text{EarlyLife} \cdot \text{Stress}$

`resultsNames``(dds)`

Name	Coefficient
Intercept	b_0
EarlyLife_ELS_vs_Standard	b_1
Stress_SVS_vs_Control	b_2
EarlyLifeELS.StressSVS	b_3

More on coefficients and the results function



Interactive session

Get your hands dirty!

Setup

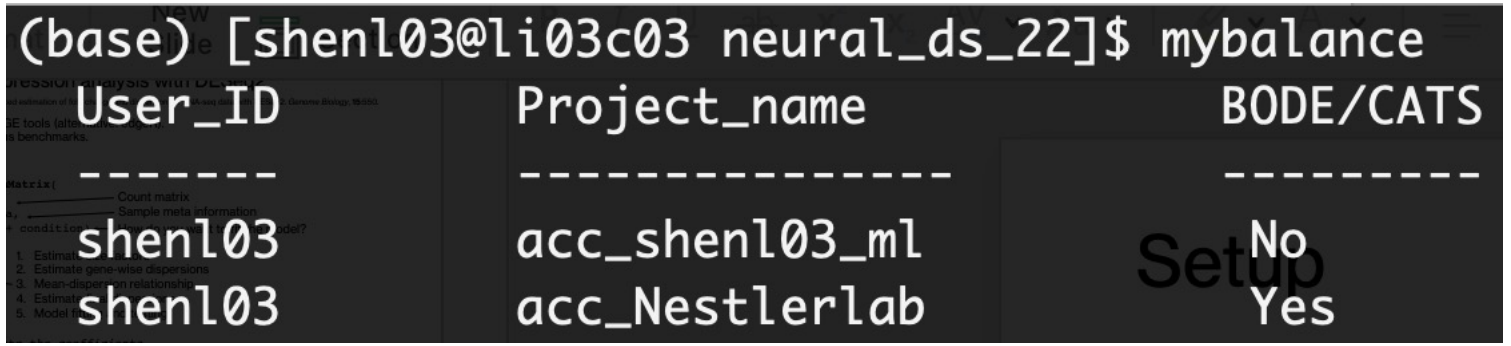
All the data used in today's course are available at:
`/sc/arion/scratch/shenl03/neural_ds_22`
on Minerva (access has been given to all participants)

Use the following command to log into a dedicated compute node for today's course: (Kudos to the Minerva team and especially Eugene Fluder)

```
bsub -U short_course -P acc_XXX -n 4 -q premium -R "rusage[mem=16G]
span[hosts=1]" -W 4:00 -Is /bin/bash
```

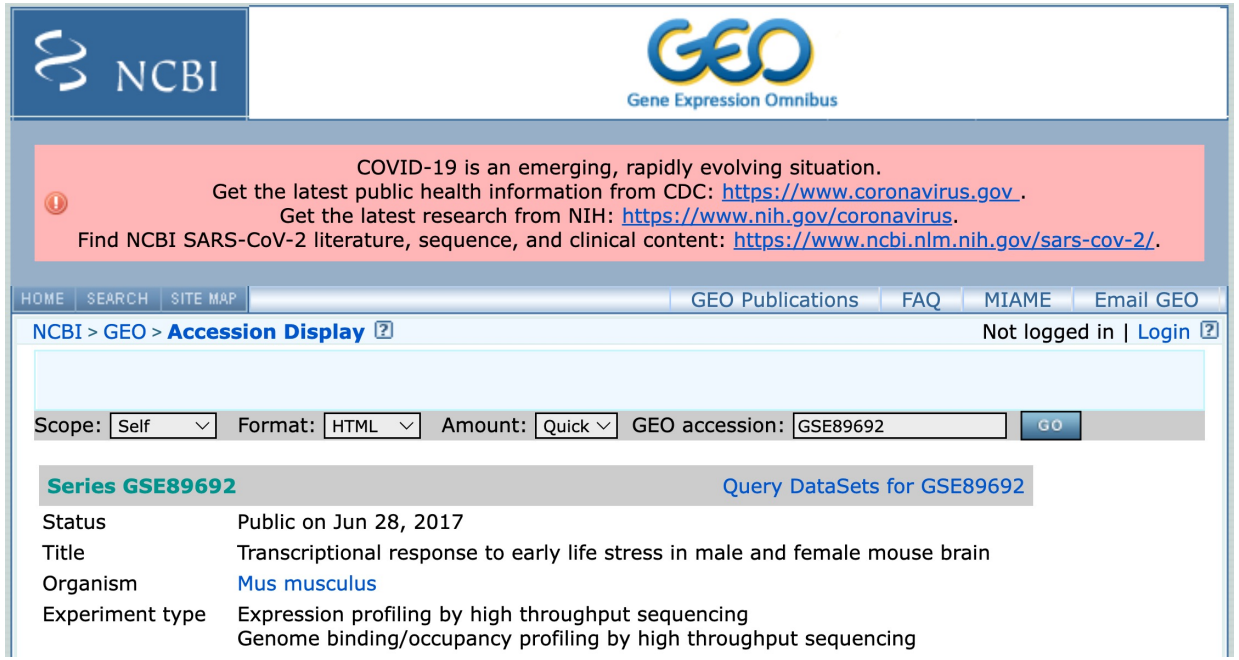
Replace this with your own account

Not sure about your account? Enter "**mybalance**" on command prompt:



Example dataset

https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE89692
GEO accession: GSE89692



Nestler EJ, Peña CJ, Purushothaman I, Loh E, Ramakrishnan A

Peña CJ, Kronman HG, Walker DM, Cates HM et al. Early life stress confers lifelong stress susceptibility in mice via ventral tegmental area OTX2. *Science* 2017 Jun 16;356(6343):1185-1188. PMID: 28619944

Peña CJ, Smith M, Ramakrishnan A, Cates HM et al. Early life stress alters transcriptomic patterning across reward circuitry in male and female mice. *Nat Commun* 2019 Nov 8;10(1):5098. PMID: 31704941

GSE89692_normalized_expression_study2_cate_modified.txt.gz	18.5 Mb	(ftp)(http)	TXT
GSE89692_rpkm.txt.gz	3.9 Mb	(ftp)(http)	TXT

SRA Run Selector

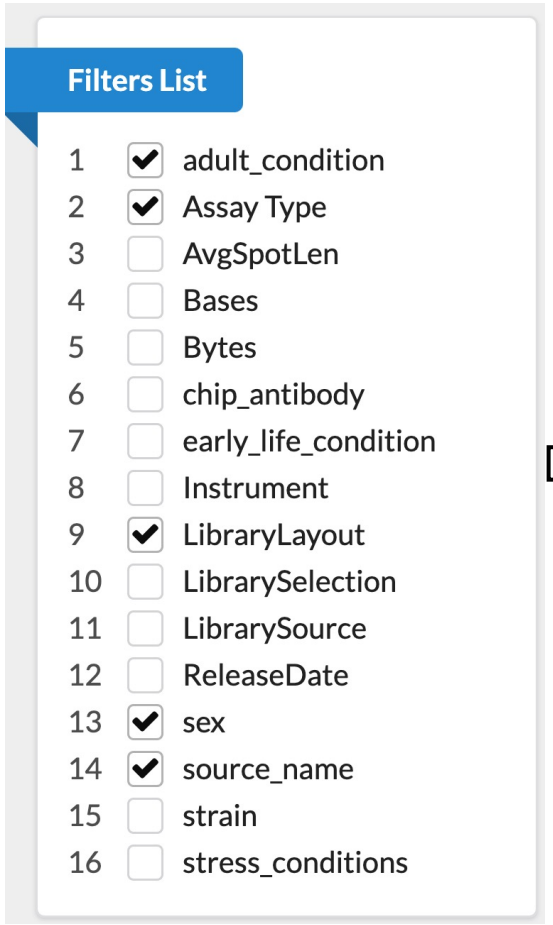
Raw data are available in SRA

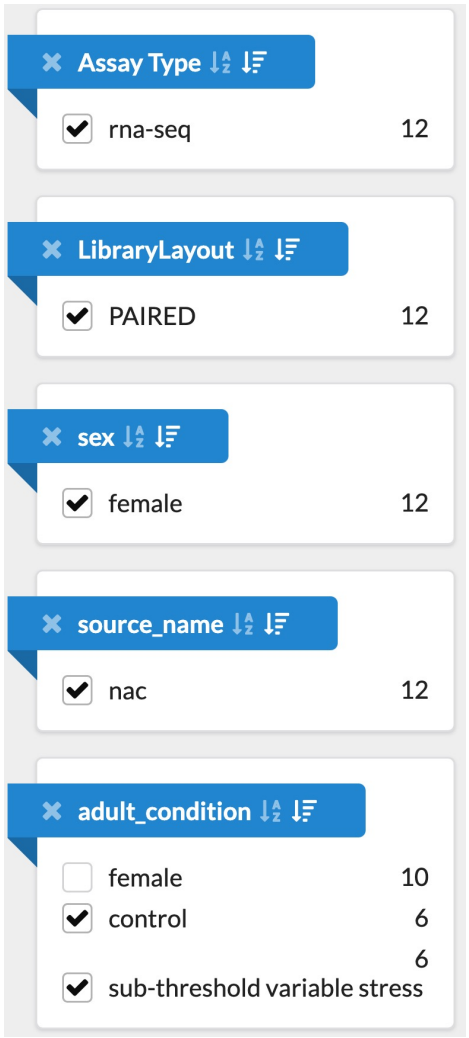
Processed data provided as supplementary file

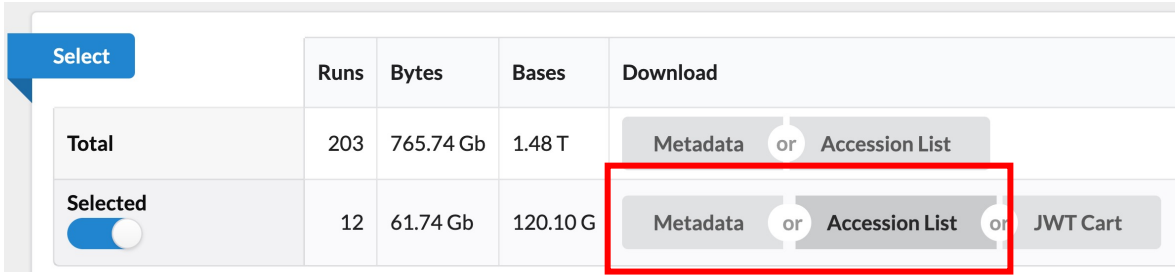
Processed data are available on Series record

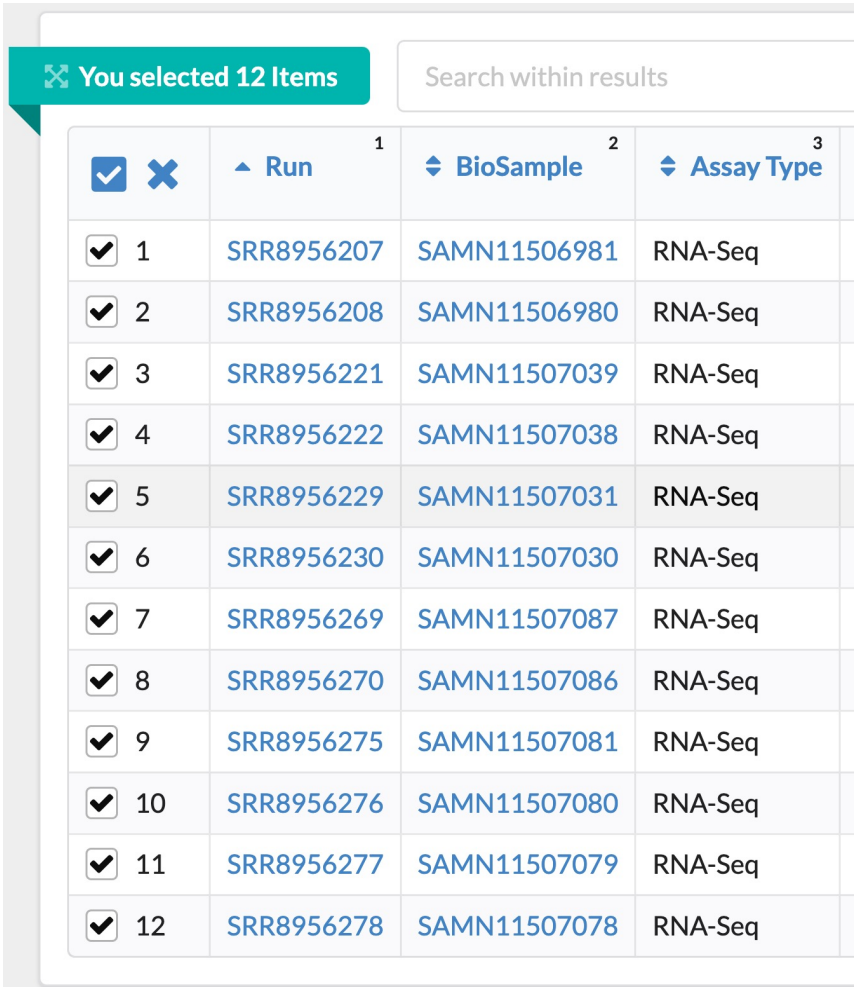
Go to the bottom of the page

SRA Run Selector









Download data from SRA

Create directory & copy the accession list:
Go to the path of your choice and `mkdir data`

```
cd /path/to/data
# scp the accession lists to the data folder.
```

Load the SRA tools: `ml sratoolkit # Minerva only.`

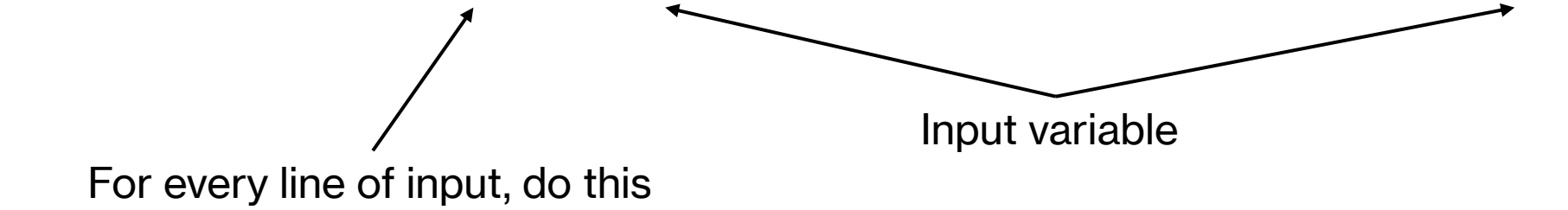
Download a single sample:

```
fastq-dump -X 1000000 --split-3 SRR8956208
```

Limit the # of reads to 1M

Download all files in one command:

```
cat srr_list.txt |xargs -I{} fastq-dump -X 1000000 --split-3 {}
```



You can directly copy the fastq files from: `/sc/arion/projects/Nestlerlab/neural_ds_22/data`

Adapter and quality trimming

Load the trim galore tool:

```
ml trim_galore
```

Test on a single sample:

```
mkdir -p ../output/trim_fastq
trim_galore --gzip --paired SRR8956208_1.fastq SRR8956208_2.fastq -o
../output/trim_fastq
```

Let's use something cooler than xargs:

```
ml parallel
```

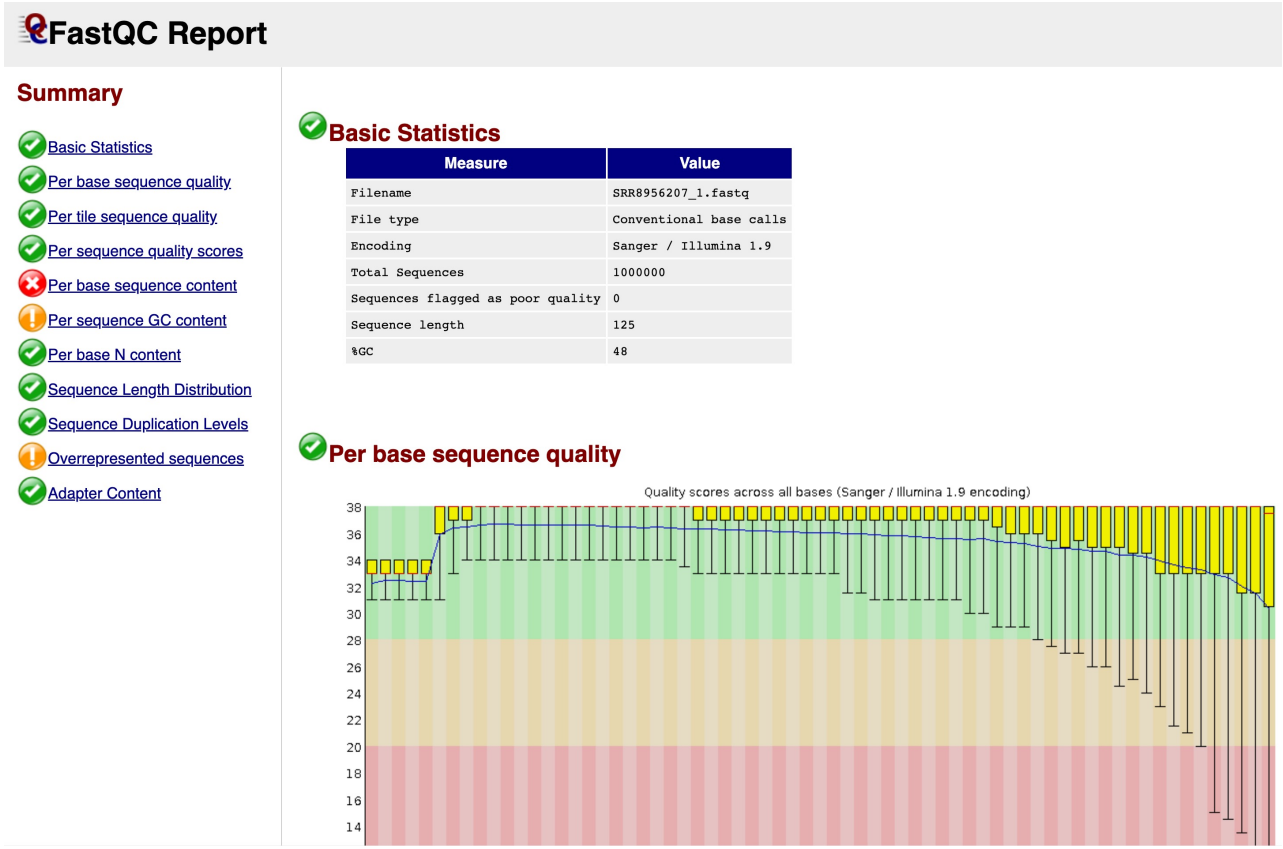
Similar to xargs but run things in parallel with additional features

```
cat srr_list.txt |parallel -j 4 "trim_galore --gzip --paired {}_1.fastq
{}_2.fastq -o ../output/trim_fastq"
```

Fastq quality assessment

```
mkdir ../output/fastqc
ml fastqc
cat srr_list.txt |parallel -j 4 "fastqc {}_1.fastq {}_2.fastq -o
../output/fastqc/"
```

You should get something like this:



Build an index on the reference sequence

```
mkdir -p genome/mouse
```

Chromosomal DNA seq

```
wget
http://ftp.ensembl.org/pub/current_fasta/mus_musculus/dna/Mus_muscul
us.GRCm39.dna_sm.chromosome.19.fa.gz
```

GTF for mouse transcriptome

```
wget
http://ftp.ensembl.org/pub/current_gtf/mus_musculus/Mus_musculus.GRCm39.105.
gtf.gz
grep "^19" Mus_musculus.GRCm39.105.gtf > Mus_musculus.GRCm39.105.chr19.gtf
```

Build index using HISAT2

```
hisat2_extract_splice_sites.py Mus_musculus.GRCm39.105.chr19.gtf >
splicesites.chr19.tsv
hisat2_extract_exons.py Mus_musculus.GRCm39.105.chr19.gtf > exons.chr19.tsv
hisat2-build -p 4 --ss splicesites.chr19.tsv --exon exons.chr19.tsv
Mus_musculus.GRCm39.dna_sm.chromosome.19.fa Mus_musculus.GRCm39.105.chr19
```

Align, sort & index

Test alignment on a single sample:

```
# go to the output folder.
mkdir bam
hisat2 -p 4 -1 trim_fastq/SRR8956207_1_val_1.fq.gz -2
trim_fastq/SRR8956207_2_val_2.fq.gz -x
../genome/mouse/Mus_musculus.GRCm39.105.chr19 -S bam/SRR8956207.sam 2>
bam/SRR8956207_hisat2.log
```

Do alignment for all samples:

```
cat ../data/srr_list.txt |parallel -j 4 "hisat2 -p 2 -1
trim_fastq/{}_1_val_1.fq.gz -2 trim_fastq/{}_2_val_2.fq.gz -x
../genome/mouse/Mus_musculus.GRCm39.105.chr19 -S bam/{}.sam 2>
bam/{}_hisat2.log"
# sort the bam files.
ls *.sam|parallel -j 4 "samtools sort -@ 2 -o {.}.bam {}"
# remove duplicates.
ls *.bam|parallel -j 4 "samtools rmdup {} {.}_rmdup.bam"
# create index.
ls *_rmdup.bam|parallel -j 4 "samtools index {}"
```

Abundance estimation

Test on a single sample:

```
mkdir counts
featureCounts -p -t gene -a ../genome/mouse/Mus_musculus.GRCm39.105.chr19.gtf
-o counts/SRR8956207_rmdup.counts.txt bam/SRR8956207_rmdup.bam 2>
counts/SRR8956207_rmdup.counts.log
```

Do counting for all samples:

```
ls bam/*_rmdup.bam|parallel -j 4 "featureCounts -p -t gene -a
../genome/mouse/Mus_musculus.GRCm39.105.chr19.gtf -o counts/{/.}.counts.txt {}
2> counts/{/.}.counts.log"
```

How do you construct a count matrix?

- **Use unix tools:** “cut” and “paste”
- **Better method:** use pandas or R
- **Best method:** use a script that can be used repeatedly

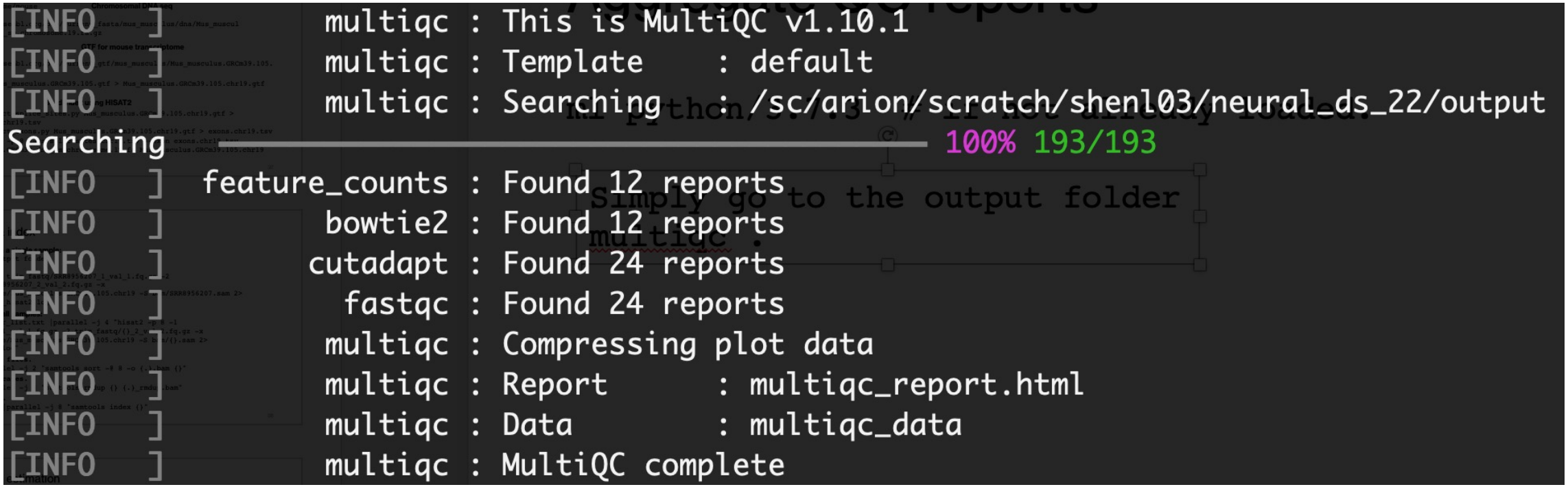
https://github.com/shenlab-sinai/NGS-Data-Charmer

```
cp /sc/arion/scratch/shenl03/neural_ds_22/NGS-Data-
Charmer/ngs_helper/generate_counts_matrix.py counts/
python generate_counts_matrix.py counts_matrix.txt *_rmdup.counts.txt
```

Aggregate QC reports

```
ml python/3.7.3  # if not already loaded.
Simply go to the output folder
multiqc .
```

You'll get some output like this:



Now, let’s switch to local!

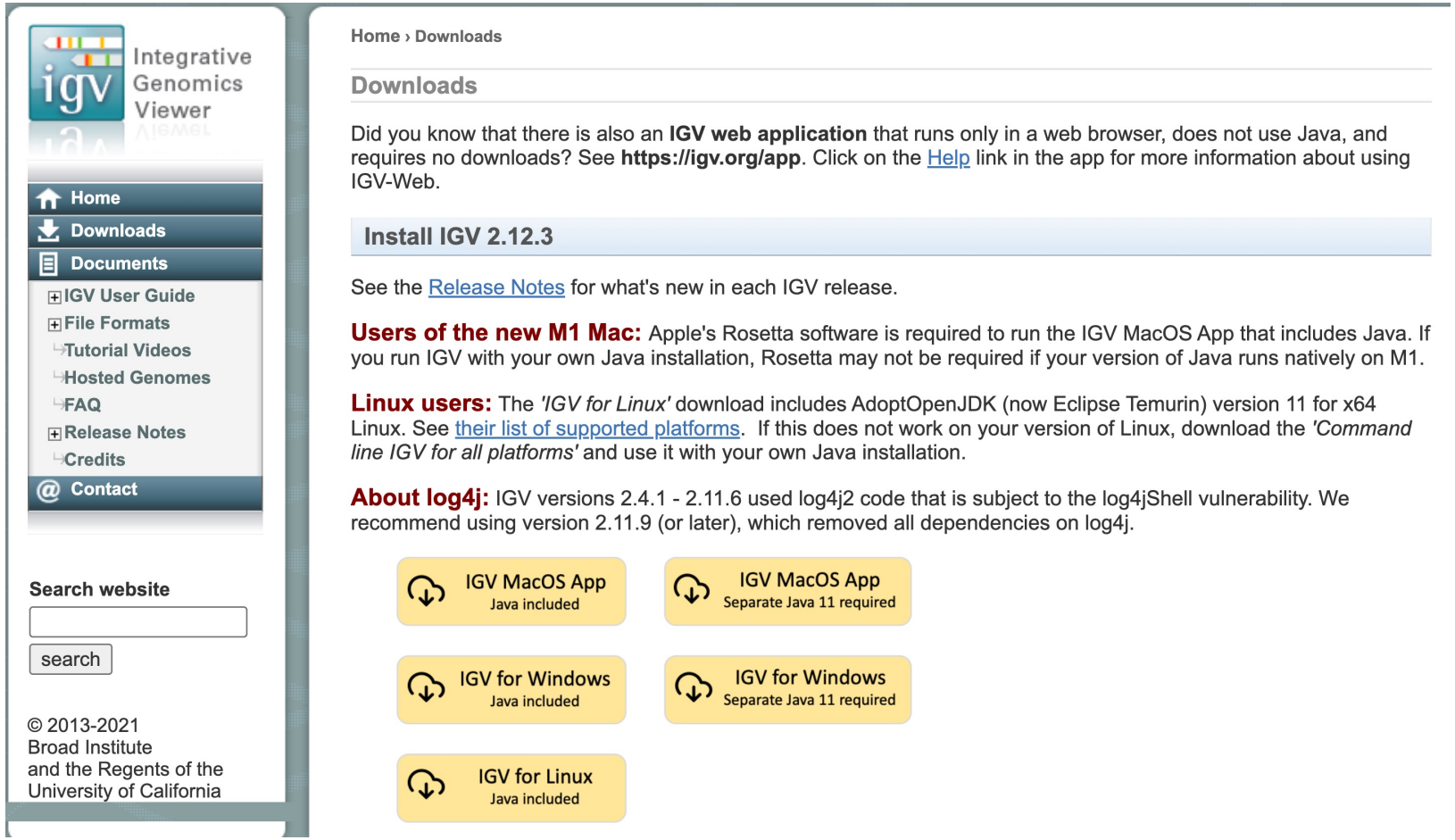
What you’ll need to download:

- The count matrix (and the meta table from SRA)
- MultiQC report
- *_rmdup.bam, *_rmdup.bam.bai

Use the IGV genome browser to view the bam files

https://software.broadinstitute.org/software/igv/download

A web version is available at: https://igv.org/app/



DGE analysis using DESeq2

https://www.bioconductor.org/packages/release/bioc/html/DESeq2.html

```
# Install DESeq2.
if (!require("BiocManager", quietly = TRUE))
        install.packages("BiocManager")
BiocManager::install("DESeq2")
```

The whole DGE R script

```
# load data.
counts_matrix <- read.delim("~/Dropbox/dump/neural_ds_22/counts_matrix.txt", row.names=1)
SraRunTable <- read.csv("~/Dropbox/dump/neural_ds_22/SraRunTable.txt")
# clean up.
colnames(counts_matrix) <- gsub('_rmdup', '', colnames(counts_matrix))
coldat <- SraRunTable[, c('Run', 'early_life_condition', 'adult_condition')]
cts <- counts_matrix[, coldat$Run]
colnames(coldat)[2:3] <- c('EarlyLife', 'Stress')
# give the conditions a better name.
coldat$EarlyLife <- gsub('ELS P10-20', 'ELS', coldat$EarlyLife)
coldat$Stress <- gsub('Sub-threshold Variable Stress', 'SVS', coldat$Stress)
# set reference level.
coldat$EarlyLife <- factor(coldat$EarlyLife, levels = c('Standard', 'ELS'))
coldat$Stress <- factor(coldat$Stress, levels = c('Control', 'SVS'))
```

DGE continued

```
# deseq2.
dds <- DESeqDataSetFromMatrix(
    countData = cts,
    colData = coldat,
    design = ~ EarlyLife + Stress + EarlyLife:Stress)
dds <- DESeq(dds)
resultsNames(dds)

# ELS effect.
els.tbl <- results(dds, name = 'EarlyLife_ELS_vs_Standard’)

# SVS effect.
svs.tbl <- results(dds, name = 'Stress_SVS_vs_Control’)

# interaction effect.
inter.tbl <- results(dds, name = 'EarlyLifeELS.StressSVS’)

# SVS effect for the ELS group.
svs.for.els.tbl <- results(dds, contrast = list(c('Stress_SVS_vs_Control',
'EarlyLifeELS.StressSVS')))
```

Visualization using variance stabilizing transformation data

```
# VST
vsd <- varianceStabilizingTransformation(dds, blind = F)

# heatmap for gene/sample matrix.
df <- as.data.frame(colData(dds)[,c("EarlyLife","Stress")])
library("pheatmap")
pheatmap(assay(vsd), cluster_rows=FALSE, show_rownames=FALSE,
         cluster_cols=T, annotation_col=df)

# sample-sample distances.
sampleDists <- dist(t(assay(vsd)))
library("RColorBrewer")
sampleDistMatrix <- as.matrix(sampleDists)
rownames(sampleDistMatrix) <- paste(vsd$EarlyLife, vsd$Stress, sep="-")
colnames(sampleDistMatrix) <- NULL
colors <- colorRampPalette( rev(brewer.pal(9, "Blues")) )(255)
pheatmap(sampleDistMatrix,
         clustering_distance_rows=sampleDists,
         clustering_distance_cols=sampleDists,
         col=colors)
```

More visualization

```
# PCA plot.
plotPCA(vsd, intgroup=c("EarlyLife", "Stress"))

# more customized PCA.
library("ggplot2")
pcaData <- plotPCA(vsd, intgroup=c("EarlyLife", "Stress"), returnData=TRUE)
percentVar <- round(100 * attr(pcaData, "percentVar"))
ggplot(pcaData, aes(PC1, PC2, color=EarlyLife, shape=Stress)) +
    geom_point(size=3) +
    xlab(paste0("PC1: ",percentVar[1],"% variance")) +
    ylab(paste0("PC2: ",percentVar[2],"% variance")) +
    coord_fixed()
```

Provide more information for the DGE list

```
# get gene symbols and full names.
BiocManager::install("AnnotationDbi")
BiocManager::install("org.Mm.eg.db")
library("org.Mm.eg.db")
inter.df <- as.data.frame(inter.tbl)
inter.df$symbol <- mapIds(org.Mm.eg.db, keys = rownames(inter.df), keytype =
"ENSEMBL", column = "SYMBOL")
inter.df$name <- mapIds(org.Mm.eg.db, keys = rownames(inter.df), keytype = "ENSEMBL",
column = "GENENAME")
```