Bioinformatic approaches to regulatory genomics and epigenomics

376-1347-00L | week 03

Pierre-Luc Germain

ETH Zürich

Plan for today

- Debriefing on the assignments
- Overview of NGS technologies
- ChIP-seq and its analysis

- Practical:
    - primary processing of a ChIP-seq experiment
      (to be continued next week)

Debriefing on the assignments: Format

- Please name the exercises just: **`assignment.html`**

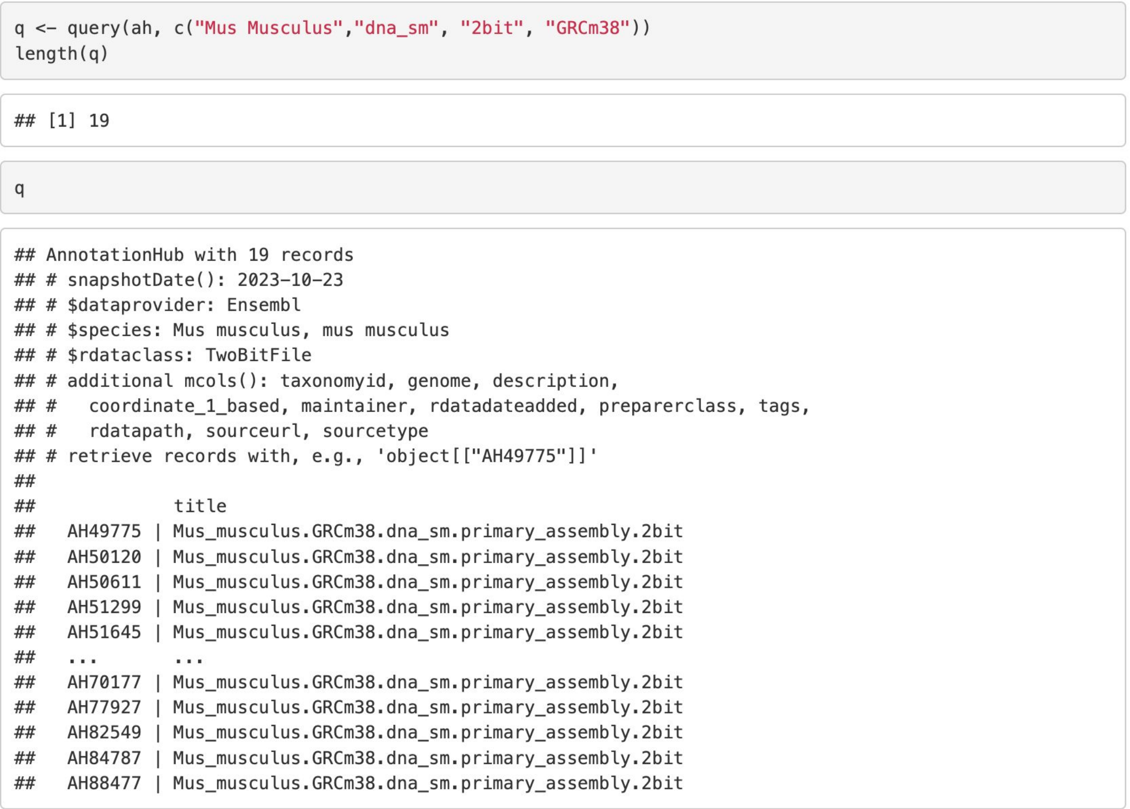
- Use titles and subtitles with **#** or **##** for the separate questions
    - e.g. for this exercise

# 1. Using AnnotationHub

## 1. a) Mouse EnsDB object

see: `https://rmarkdown.rstudio.com/authoring_basics.html`

Debriefing on the assignments: Date added



```r
q <- query(ah, c("Mus Musculus","dna_sm", "2bit", "GRCm38"))
length(q)
```

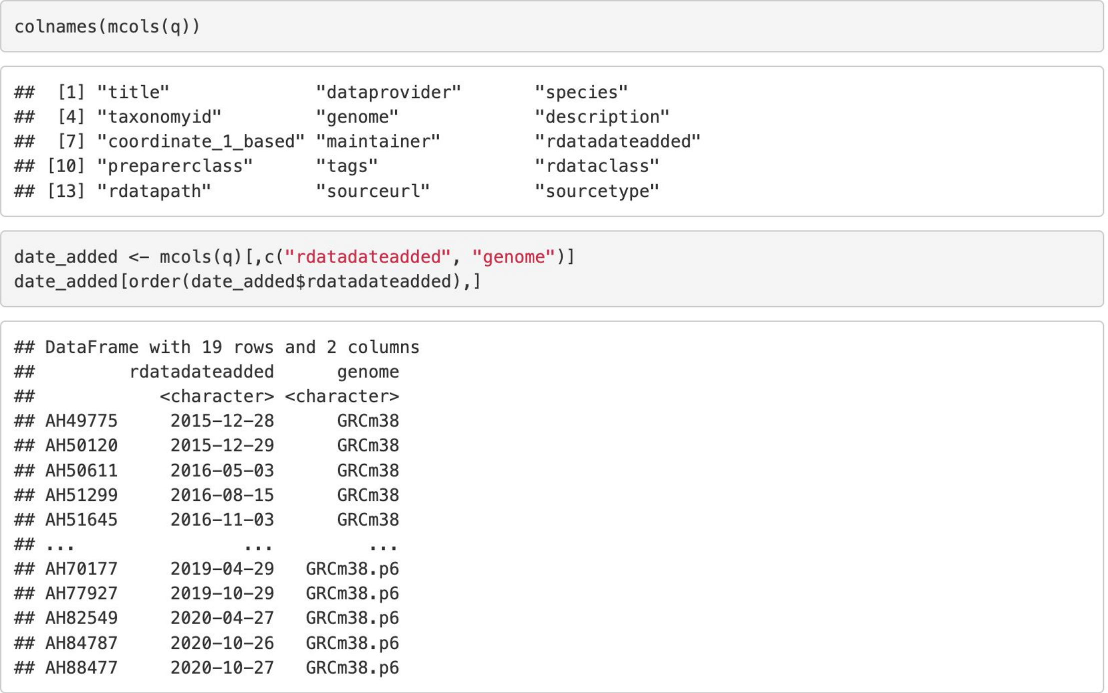
```
## [1] 19
```

```r
q
```

```
## AnnotationHub with 19 records
## # snapshotDate(): 2023-10-23
## # $dataprovider: Ensembl
## # $species: Mus musculus, mus musculus
## # $rdataclass: TwoBitFile
## # additional mcols(): taxonomyid, genome, description,
## #   coordinate_1_based, maintainer, rdatadateadded, preparerclass, tags,
## #   rdatapath, sourceurl, sourcetype
## # retrieve records with, e.g., 'object[["AH49775"]]'
##
##             title
##   AH49775 | Mus_musculus.GRCm38.dna_sm.primary_assembly.2bit
##   AH50120 | Mus_musculus.GRCm38.dna_sm.primary_assembly.2bit
##   AH50611 | Mus_musculus.GRCm38.dna_sm.primary_assembly.2bit
##   AH51299 | Mus_musculus.GRCm38.dna_sm.primary_assembly.2bit
##   AH51645 | Mus_musculus.GRCm38.dna_sm.primary_assembly.2bit
##   ...       ...
##   AH70177 | Mus_musculus.GRCm38.dna_sm.primary_assembly.2bit
##   AH77927 | Mus_musculus.GRCm38.dna_sm.primary_assembly.2bit
##   AH82549 | Mus_musculus.GRCm38.dna_sm.primary_assembly.2bit
##   AH84787 | Mus_musculus.GRCm38.dna_sm.primary_assembly.2bit
##   AH88477 | Mus_musculus.GRCm38.dna_sm.primary_assembly.2bit
```

Debriefing on the assignments: Date added

- If several results are obtained from a query, one can look at the metadata with:



```
colnames(mcols(q))
```

```
##  [1] "title"              "dataprovider"       "species"
##  [4] "taxonomyid"         "genome"             "description"
##  [7] "coordinate_1_based" "maintainer"         "rdatadateadded"
## [10] "preparerclass"      "tags"               "rdataclass"
## [13] "rdatapath"          "sourceurl"          "sourcetype"
```

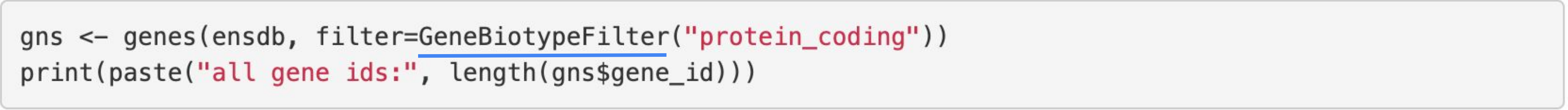
```
date_added <- mcols(q)[,c("rdatadateadded", "genome")]
date_added[order(date_added$rdatadateadded),]
```

```
## DataFrame with 19 rows and 2 columns
##         rdatadateadded      genome
##            <character> <character>
## AH49775     2015-12-28      GRCm38
## AH50120     2015-12-29      GRCm38
## AH50611     2016-05-03      GRCm38
## AH51299     2016-08-15      GRCm38
## AH51645     2016-11-03      GRCm38
## ...                ...         ...
## AH70177     2019-04-29   GRCm38.p6
## AH77927     2019-10-29   GRCm38.p6
## AH82549     2020-04-27   GRCm38.p6
## AH84787     2020-10-26   GRCm38.p6
## AH88477     2020-10-27   GRCm38.p6
```

Debriefing on the assignments: Using filters

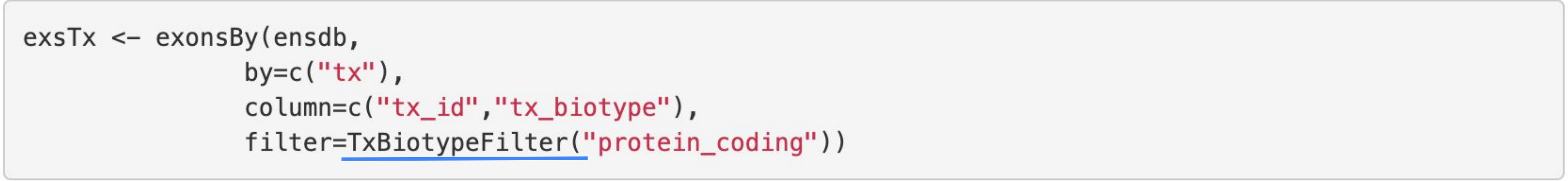
We can use filters directly when retrieving annotations from an `EnsDb` object.

- 2.1:



```
gns <- genes(ensdb, filter=GeneBiotypeFilter("protein_coding"))
print(paste("all gene ids:", length(gns$gene_id)))
```

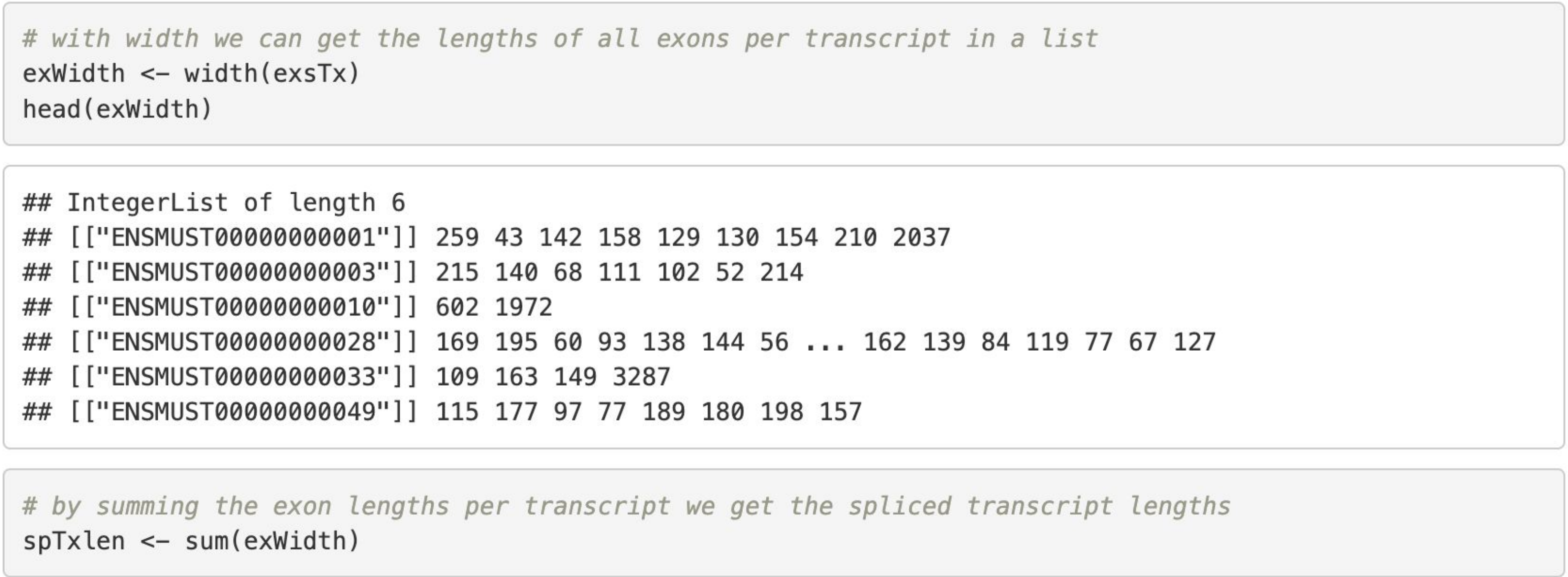
- 2.2:



```
exsTx <- exonsBy(ensdb,
                by=c("tx"),
                column=c("tx_id","tx_biotype"),
                filter=TxBiotypeFilter("protein_coding"))
```

Debriefing on the assignments: Getting lengths of spliced transcripts

2.2:



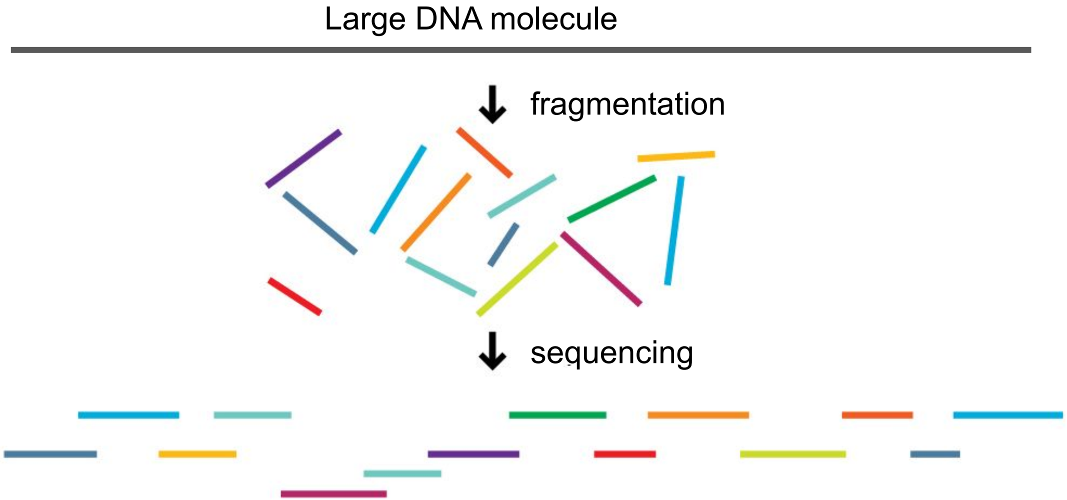
```
# with width we can get the lengths of all exons per transcript in a list
exWidth <- width(exsTx)
head(exWidth)
```

```
## IntegerList of length 6
## [["ENSMUST00000000001"]] 259 43 142 158 129 130 154 210 2037
## [["ENSMUST00000000003"]] 215 140 68 111 102 52 214
## [["ENSMUST00000000010"]] 602 1972
## [["ENSMUST00000000028"]] 169 195 60 93 138 144 56 ... 162 139 84 119 77 67 127
## [["ENSMUST00000000033"]] 109 163 149 3287
## [["ENSMUST00000000049"]] 115 177 97 77 189 180 198 157
```

```
# by summing the exon lengths per transcript we get the spliced transcript lengths
spTxlen <- sum(exWidth)
```

Next Generation Sequencing (NGS)

Shotgun sequencing:



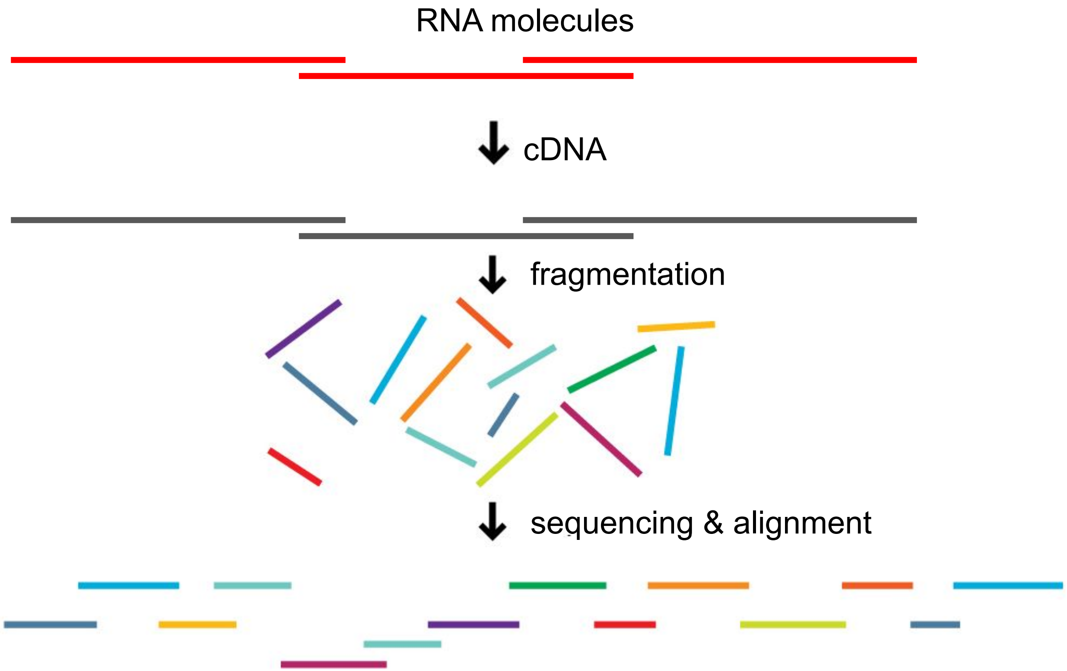
Large DNA molecule

fragmentation

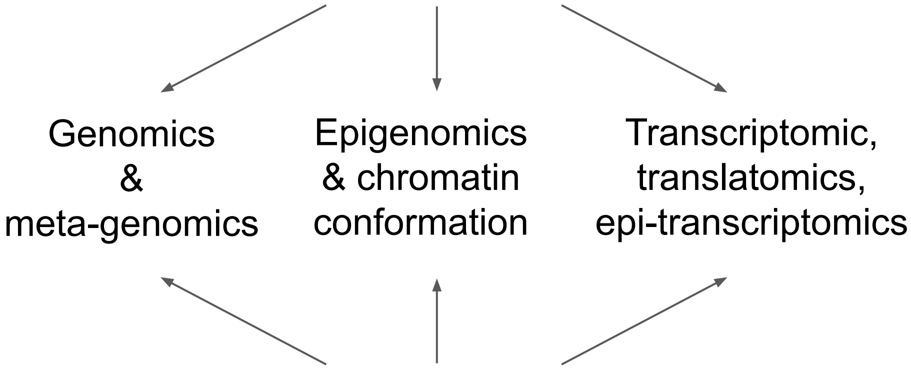
sequencing

Next Generation Sequencing (NGS)

RNA sequencing:



Next Generation Sequencing:
one technology to rule them all

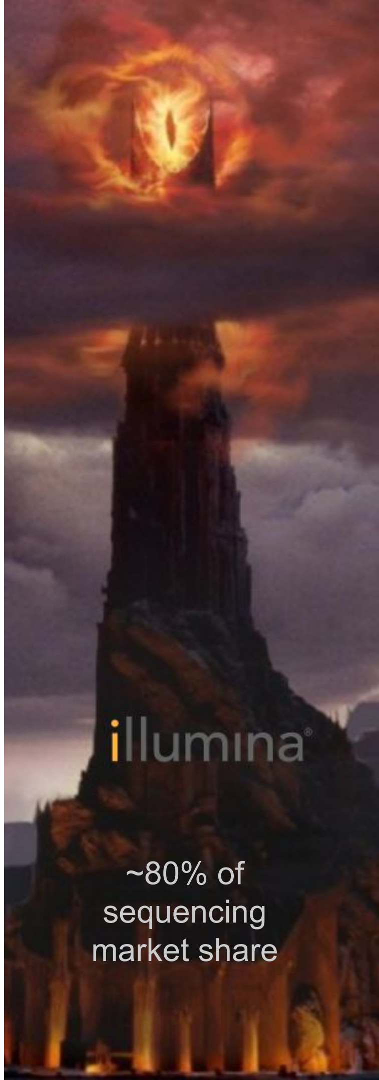


Genomics
&
meta-genomics

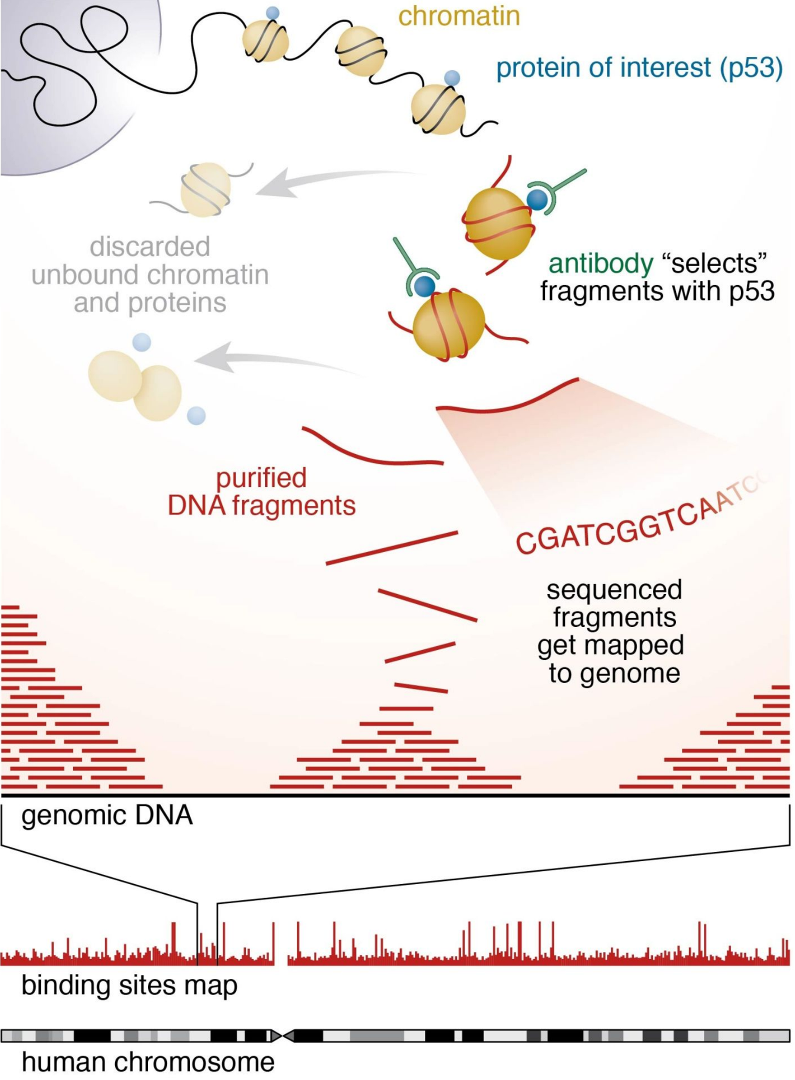
Epigenomics
& chromatin
conformation

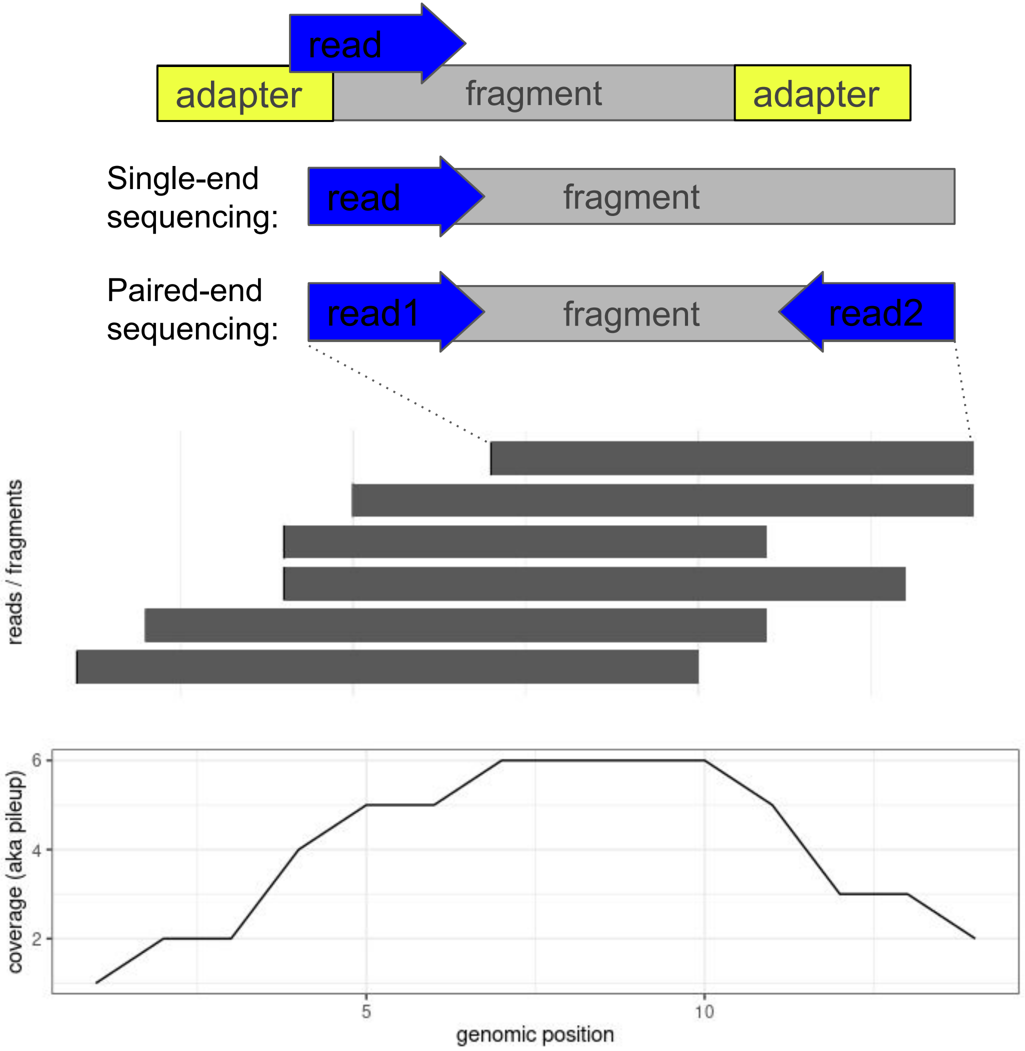
Transcriptomic,
translatomics,
epi-transcriptomics

A lot of convergence in terms of analysis
tools and techniques

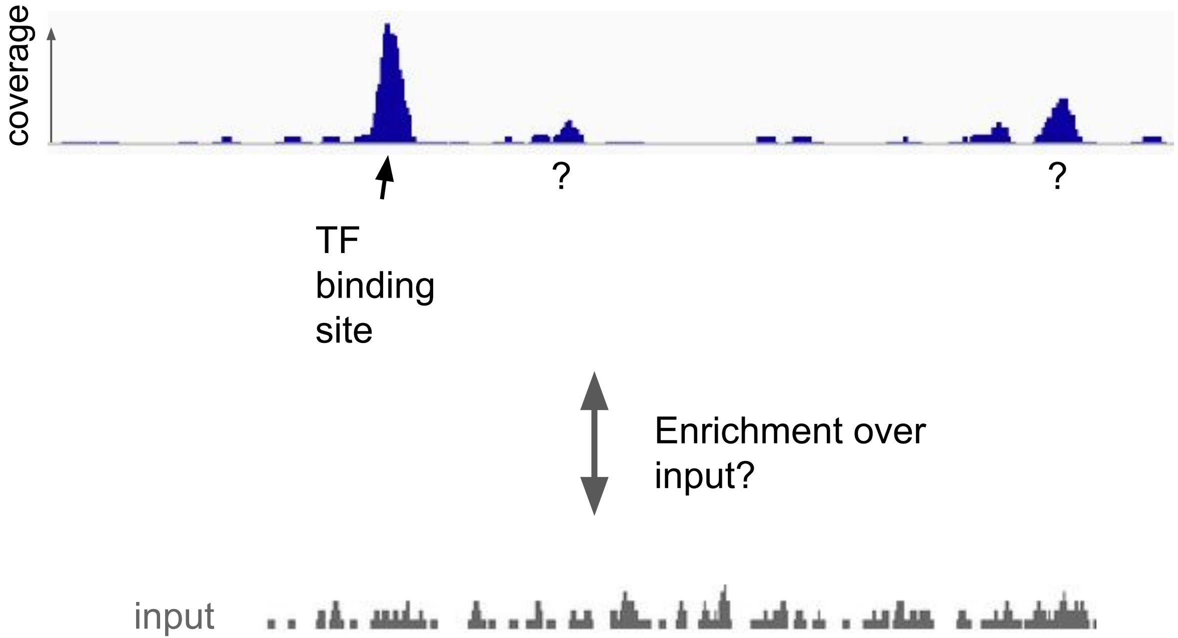


~80% of
sequencing
market share


chromatin
protein of interest (p53)
discarded
unbound chromatin
and proteins
antibody “selects”
fragments with p53
purified
DNA fragments
CGATCGGTCAATC
sequenced
fragments
get mapped
to genome
genomic DNA
binding sites map
human chromosome


read
adapter
fragment
adapter
Single-end
sequencing:
read
fragment
Paired-end
sequencing:
read1
fragment
read2
reads / fragments
coverage (aka pileup)
6
4
2
5
10
genomic position

Peak calling

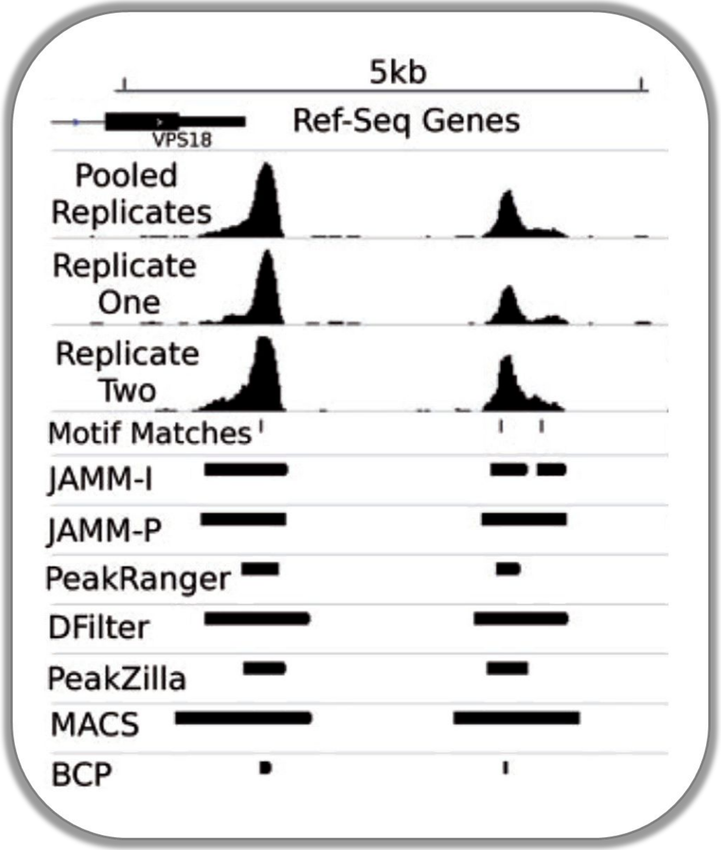


TF binding site

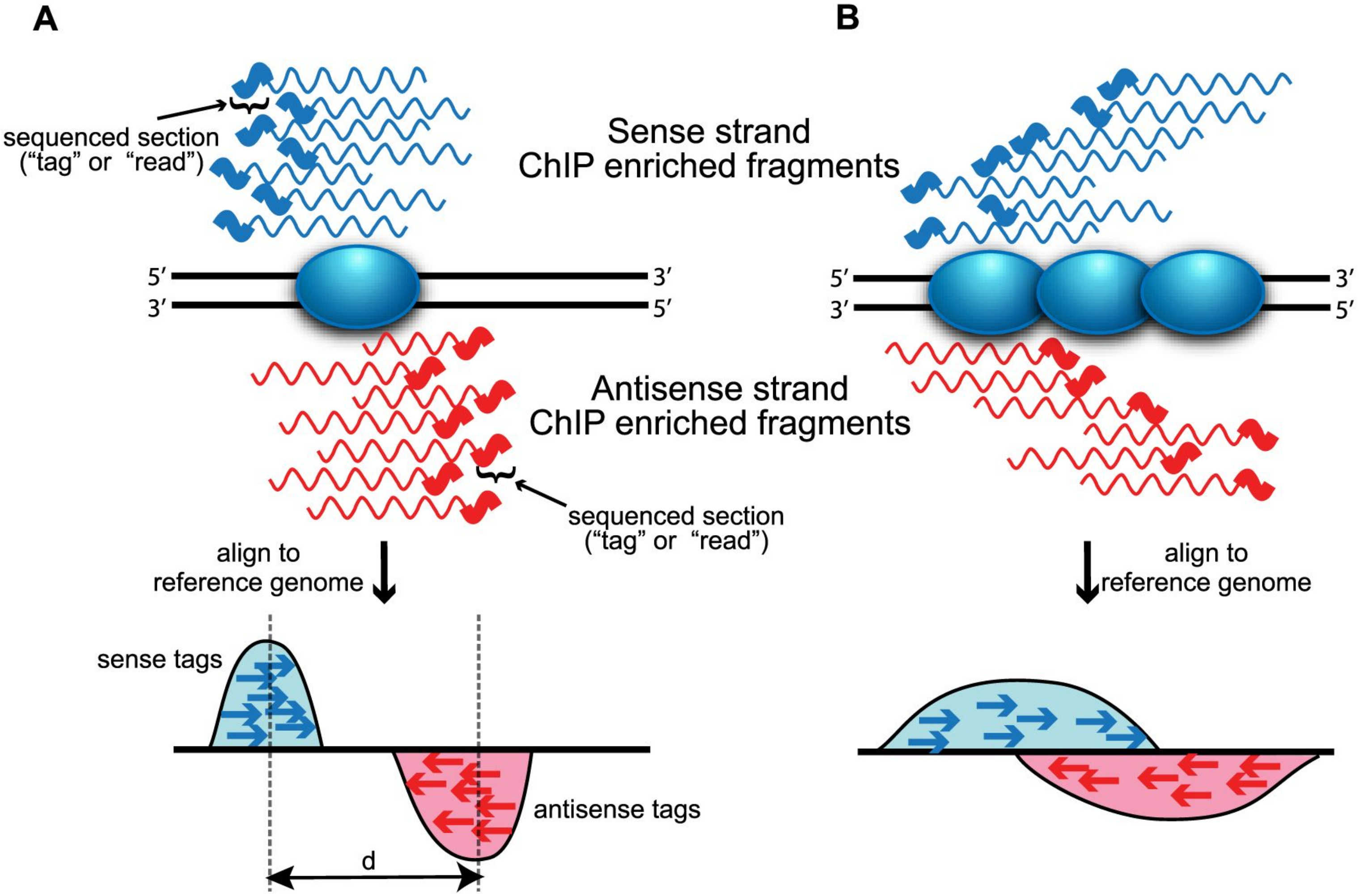
?

?

Enrichment over input?

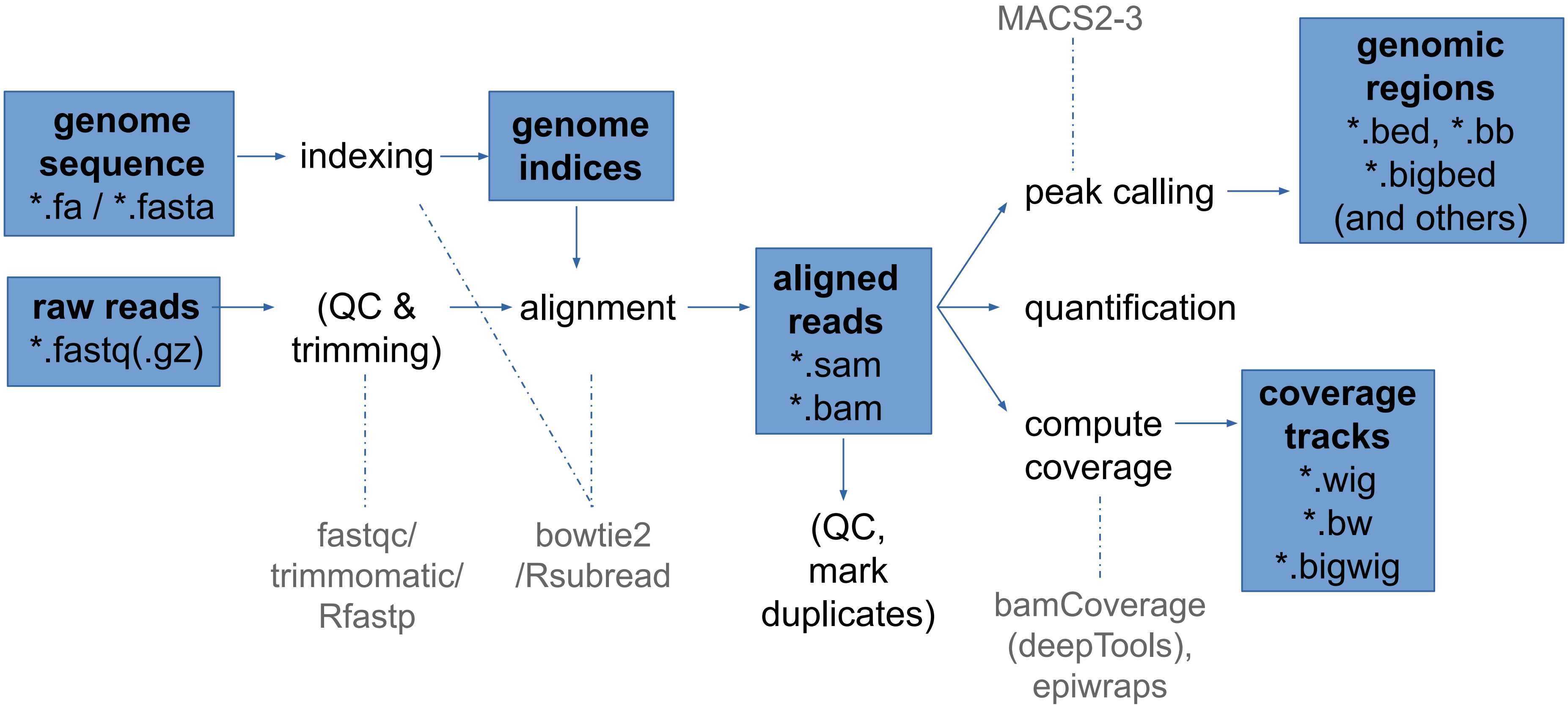


(Ibrahim et al., NAR 2014)



(Wilbanks et al., PLoS One 2010)

Overview of a primary analysis pipeline (ChIP-seq and the likes)


genome sequence *.fa / *.fasta
indexing
genome indices
raw reads *.fastq(.gz)
(QC & trimming)
alignment
aligned reads *.sam *.bam
MACS2-3
peak calling
genomic regions *.bed, *.bb *.bigbed (and others)
quantification
compute coverage
coverage tracks *.wig *.bw *.bigwig
fastqc/ trimmomatic/ Rfastp
bowtie2 /Rsubread
(QC, mark duplicates)
bamCoverage (deepTools), epiwraps

Alternative toolsets for (DNA) primary analysis

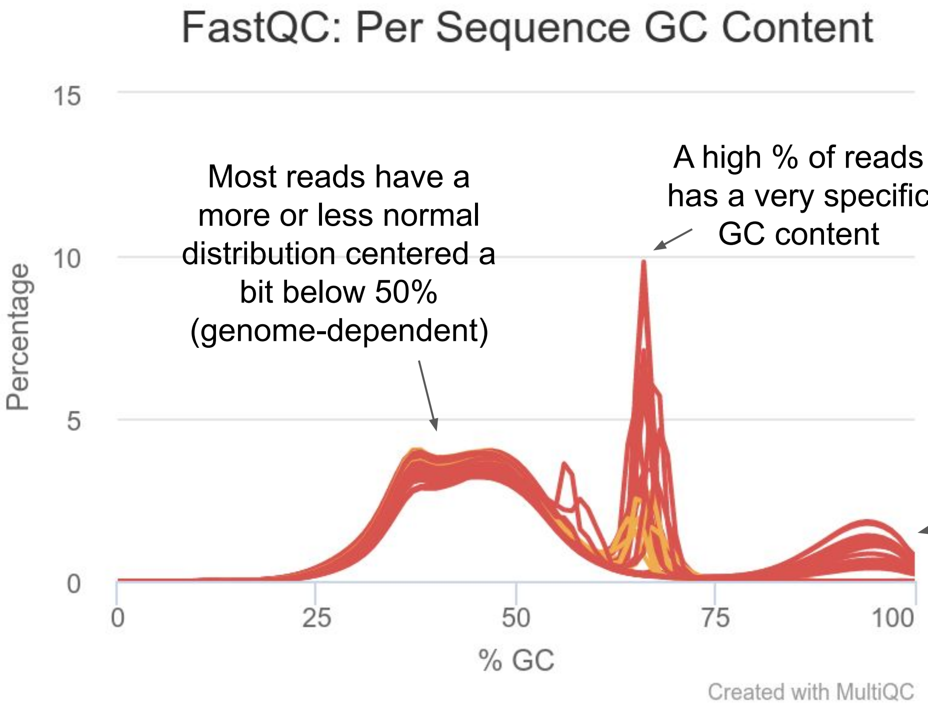
- The most standard one:
  - fastqc
  - trimmomatic
  - bowtie2
  - picard
  - deeptools

- Pure R-based
  - rfastp QuasR
  - Rsubread

Downstream analysis (R)

- epiwraps
- ChIPseeker
- etc…

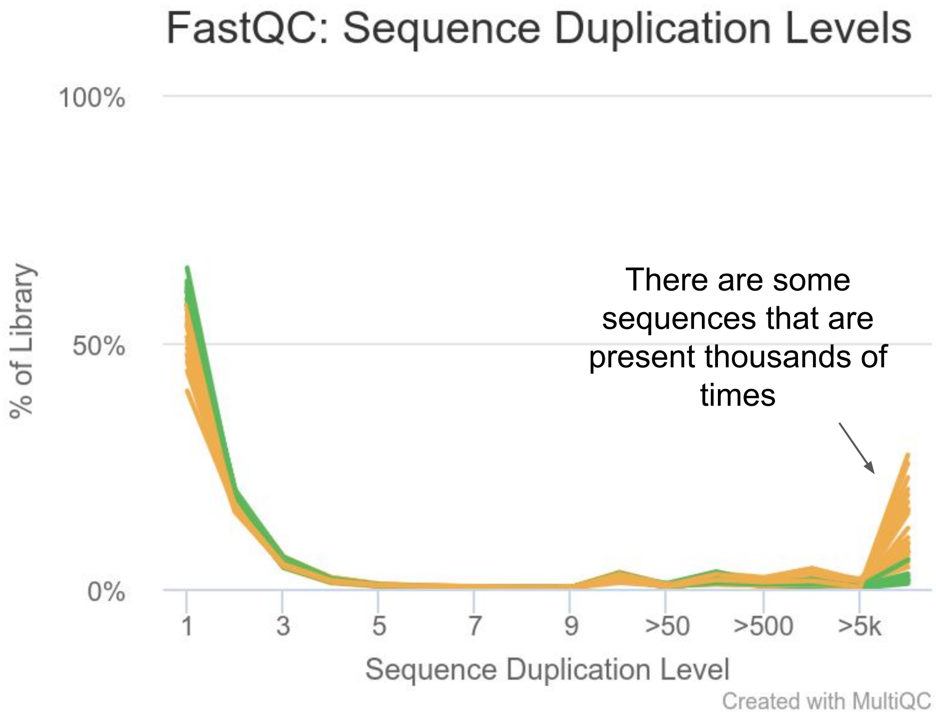
Example (rather extreme) QC problems


FastQC: Per Sequence GC Content

Most reads have a more or less normal distribution centered a bit below 50% (genome-dependent)

A high % of reads has a very specific GC content

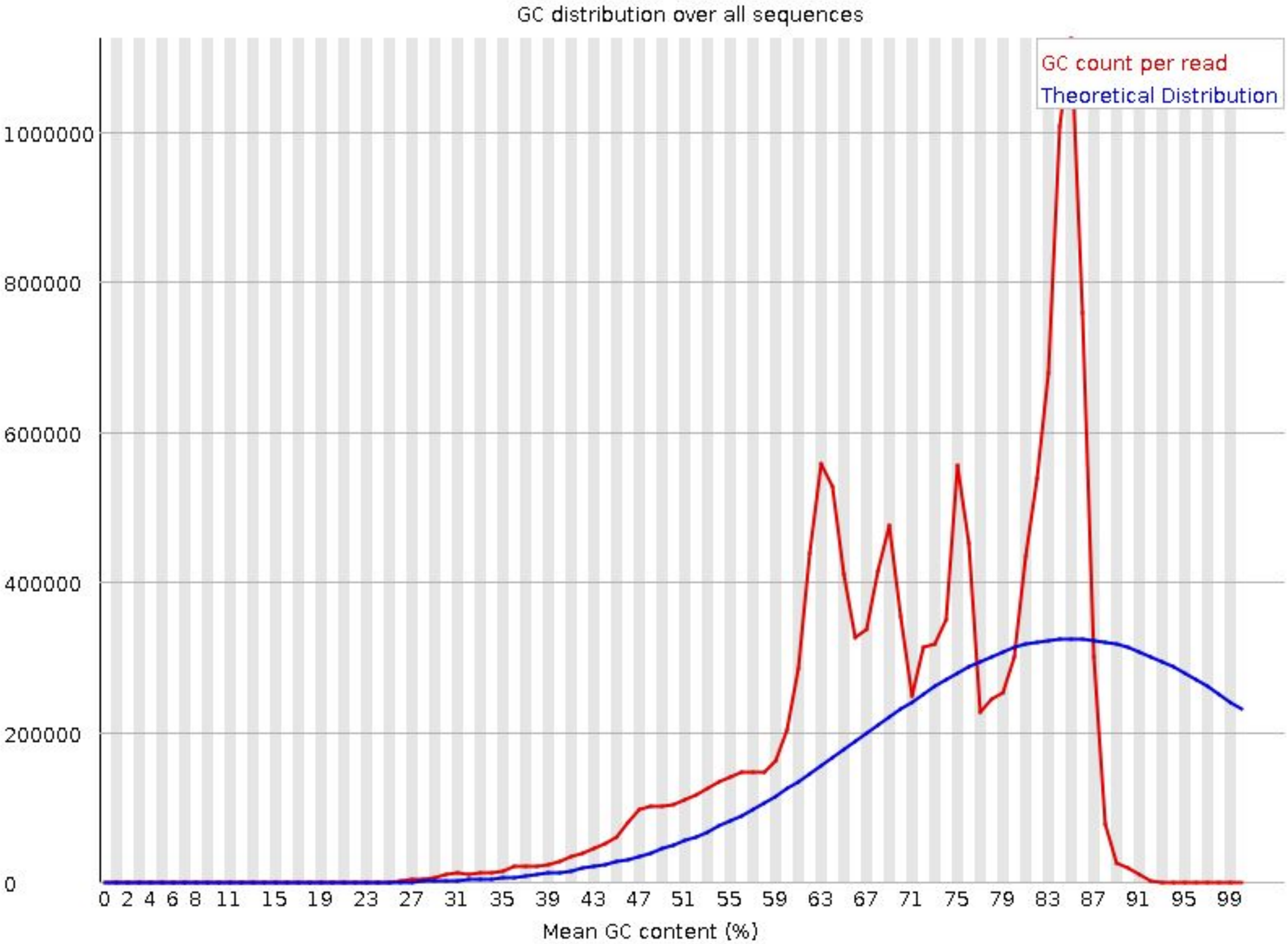
A certain % of the reads has an extremely high GC content


FastQC: Sequence Duplication Levels

There are some sequences that are present thousands of times

Example (rather extreme) QC problems:

Bias from overamplification

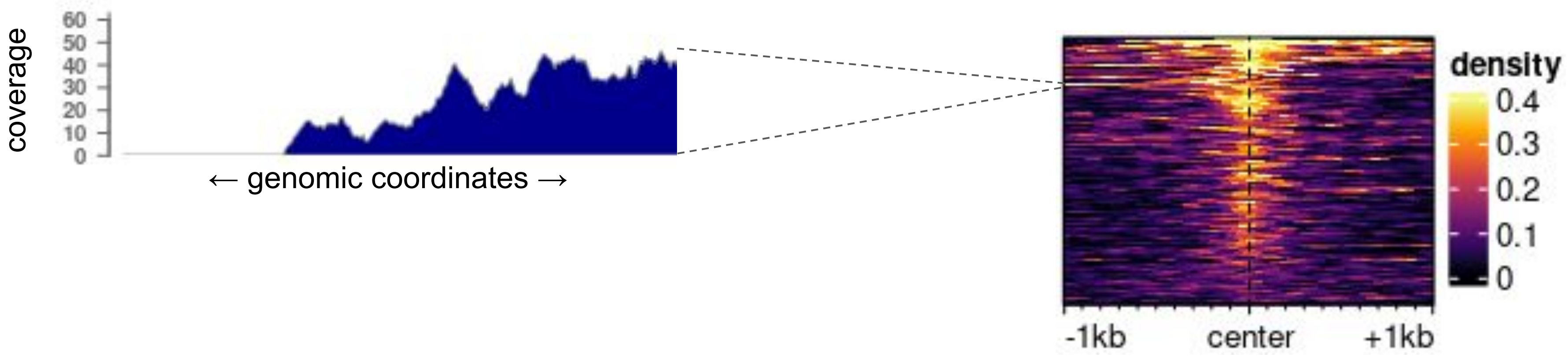


Visualizations available in *epiwraps*

Documentation

- Signal across one genomic region: `plotSignalTracks`

- Signal across several genomic regions: `signal2Matrix` → `plotEnrichedHeatmaps`



(Based on the *Gviz* R package)

(Mainly based on the EnrichedHeatmap R package, itself based on ComplexHeatmap)

Assignment

- Download the following Drosophila ChIP-seq for the protein CTCF:
  - IP: https://www.encodeproject.org/files/ENCFF127RRR/@@download/ENCFF127RRR.fastq.gz

    (no input control for the purpose of this exercise)
- Process it from the raw data, obtaining:
  - bam file
  - peaks
- Report:
  - how many reads (and what percentage) were mapped
  - how many peaks were found
- Plot the signal around one of the peaks that is located *inside a gene*

- Please make sure that you name your final file **`assignment.html`** !!