# STAT115 Lab 6: ChIP-Seq & ATAC-Seq

TA: Jiazhen Rong

2020.03.02

*Some Slide Content Adapted from Previous Year's TA Qian Xiao

# Several Announcements

- Please submit both .html and .Rmd for HW submissions. (HW2 - if missing .html, please submit before Thursday).
- HW3 is due on Mon 3/8, 2021 @ 11:59pm
- Check #hw3 and #homework-questions channel in Slack for HW3
- OH on Friday 4-5pm and Sat 10:30-11:30AM

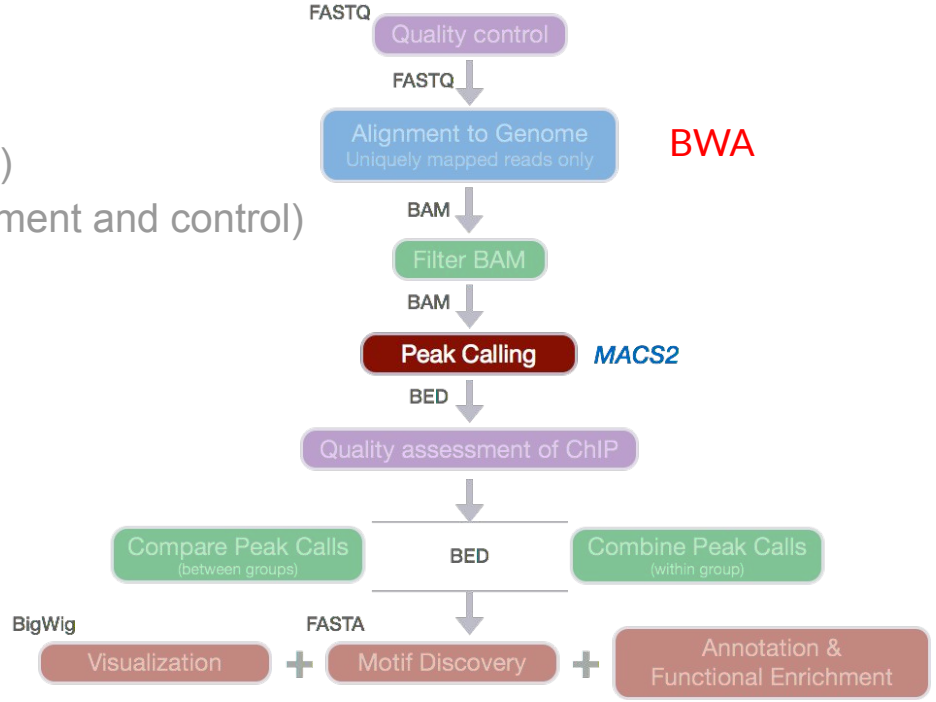
# HW3 Clarifications

- Undergrads do not need to run Q2, because MACS can directly take .bam files as well in Q3. Undergrads do not need to compare Q9 result with Q8 (grad question).
- For Q2, please use the provided tumor.bam and normal.bam, but not the .fastq file.
- We can see mouse TFs enriched as well (e.g. human - FOXA2, mouse - Foxa2). It is okay to use these TFs.
- For most downstream questions that require ‘differential’ AR ChIP peaks, they refer to Q4 output.

# Recap of ChIP-Seq Analysis Workflow

1. Align reads (Input Fastq output BAM)
2. Remove duplicates (Input BAM output BED)
3. Downsample (balance reads between treatment and control)
4. Call peaks (Input BAM output BED)
5. Visualize peaks (UCSC)
6. Motif and Co-interacting TF Finding
7. **Integration with gene expression data**

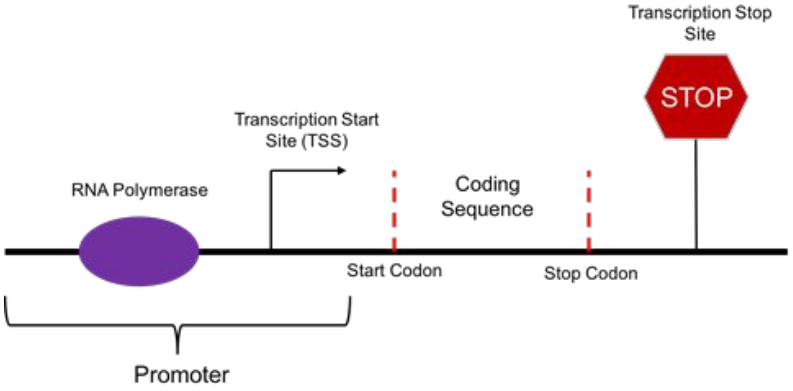
……



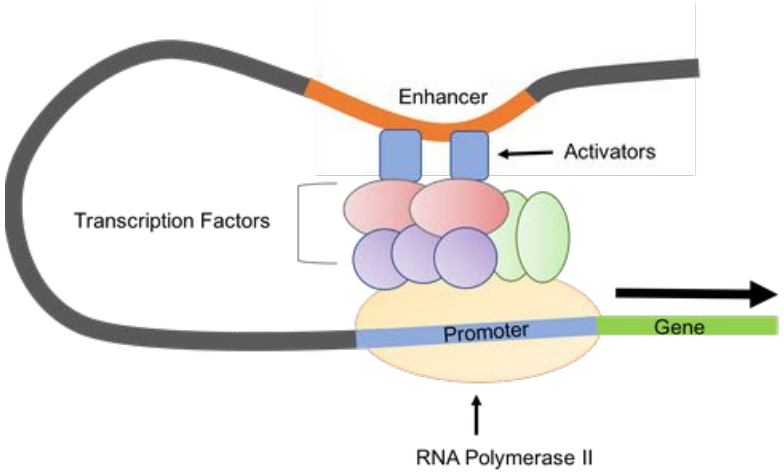

**Picture Credit: https://hbctraining.github.io/Intro-to-ChIPseq/lessons/05_peak_calling_macs.html**

# Integrate ChIP-Seq with Differentially Expressed Genes (DEG)
# (Part V of HW3)

# Gene Regulation




Gene Transcription w/o any TF

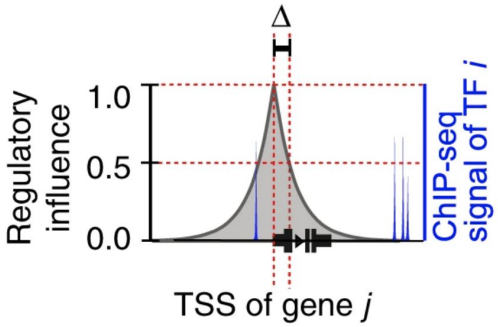



Gene Transcription with TF

Promoter Region: 100-1000bp long before TSS
Enhancer Region: Could be much further away

(Addgene, 2021)

# Regulatory Potential (RP) of TF




$$R_{i,j}(\Delta) = \sum_{peak-k-of-TF_i} 2^{\frac{-X_{ijk}}{\Delta}}$$

The regulatory effect of TF *i* on gene *j* is modeled as the RP, R*i, j*(Δ)

(Chen et.al, Nature Communications, 2020)

- Distance-based Model
- Exponential decay of RP as distance to TSS increase
- The provided regulatory score file in HW3 is calculated by BETA

Original Paper about BETA

# BED file and BEDTools

```
The bedtools sub-commands include:

[ Genome arithmetic ]
    intersect     Find overlapping intervals in various ways.
    window        Find overlapping intervals within a window around an interval.
    closest       Find the closest, potentially non-overlapping interval.
    coverage      Compute the coverage over defined intervals.
    map           Apply a function to a column for each overlapping interval.
```

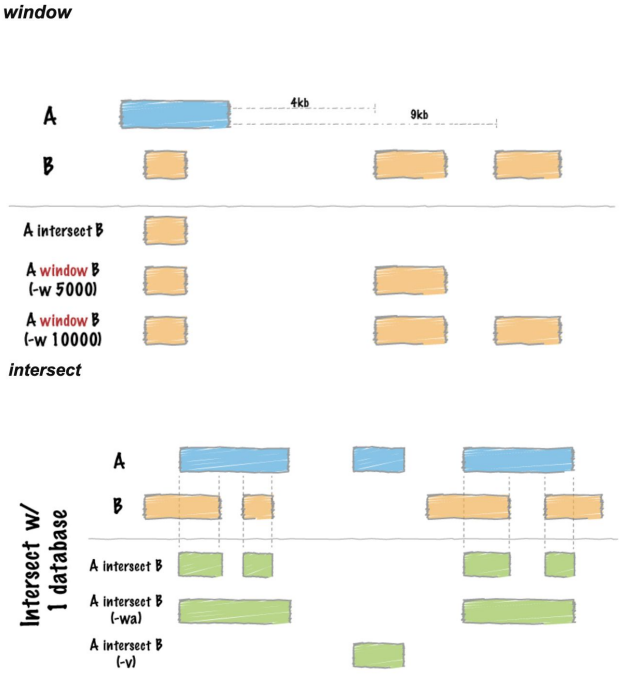
Format of .bed file:

**chrom chromeStart chromEnd** name score **strand** .....

| | | | | | |
|---|---|---|---|---|---|
| chr7 | 127471196 | 127472363 | Pos1 | 0 | + |
| chr7 | 127472363 | 127473530 | Pos2 | 0 | + |
| chr7 | 127473530 | 127474697 | Pos3 | 0 | + |
| chr7 | 127474697 | 127475864 | Pos4 | 0 | + |

**BEDTools** (developed in 2010)
Official Tutorial, Documentation

Some popular commands:
*bedtools intersect*
*bedtools overlap*
***bedtools subtract***
***bedtools window***
.....



# Q12: For Graduate Students Only

Now we try to see what direct target genes these AR binding sites regulate. Among the differentially expressed genes in prostate cancer, only a subset might be directly regulated by AR binding. In addition, among all the genes with nearby AR binding, only a subset might be differentially expressed. One simple way of getting the **AR target genes** is to look at **which genes have AR binding in its promoters**. Write a python program that takes two input files: 1) the AR differential ChIP-seq peaks in tumor over normal; 2) refGene annotation. The program outputs to a file containing genes that have AR ChIP-seq peak (in this case, stronger peak in tumor) within **2KB** +/- from the transcription start site (TSS) of each gene. **How many putative AR target genes** in prostate cancer do you get using this approach?

Hint:

1) The RefGene annotation table is already in **/n/stat115/2021/HW3/refGene.txt** in Cannon.
2) TSS is different for genes on positive or negative strand, i.e. TSS is “txStart” for genes on the positive strand, “txEnd” for genes in negative strand. When testing your python code, try smaller number of gene annotations or smaller number of peaks to check your results before moving forward.
3) Instead of writing the whole process in Python code (which might take a long time to run), you can rewrite TSS starting positions of refGene based on strand in Python, and then perform the 2KB +/- check by command line tool BEDTools (https://bedtools.readthedocs.io/en/latest/).

## What is within RefGene?

-- An annotation of human genes

```
585     NR_106918       chr1    -       17368   17436   17436   17436   1       17368,  17436,  0       MIR6859-1       unk     unk     -1,
585     NR_107062       chr1    -       17368   17436   17436   17436   1       17368,  17436,  0       MIR6859-2       unk     unk     -1,
585     NR_107063       chr1    -       17368   17436   17436   17436   1       17368,  17436,  0       MIR6859-3       unk     unk     -1,
585     NR_128720       chr1    -       17368   17436   17436   17436   1       17368,  17436,  0       MIR6859-4       unk     unk     -1,
585     NR_036051       chr1    +       30365   30503   30503   30503   1       30365,  30503,  0       MIR1302-2       unk     unk     -1,
```

(You can view the content of a file by `less file_name` and `q` to quit viewing)

# Q12: For Graduate Students Only

**Re-writing RefGene Annotation to BED format:**

```python
import pandas as pd

#Read in the reference data
TSS = pd.read_table('/path/to/refGene.txt', header=None)

#Create a subset from it with the columns we need
TSS_sub = TSS.iloc[:,[2,4,5,3,12]]
TSS_sub.columns = ["chr","start","end","strand","id"]

#Create subsets of genes on positive strand and negative strand
TSS_pos = TSS_sub[TSS_sub.strand == '+']
TSS_neg = TSS_sub[TSS_sub.strand == '-']

#Rearrange the data and create a new dataframe containing the information of TSS we want
TSS_pos.end = TSS_pos.start
TSS_neg.start = TSS_neg.end
TSS_new = TSS_pos.append(TSS_neg)
TSS_new.end = TSS_new.end + 1

#Export it into a bed file
TSS_new.to_csv("/your/TSS_hg38_bed_file",sep='\t',header=False, index=False)
```

Use python to re-write refGene.txt into BED format based on strand
(Live demo in Jupyter Notebook)

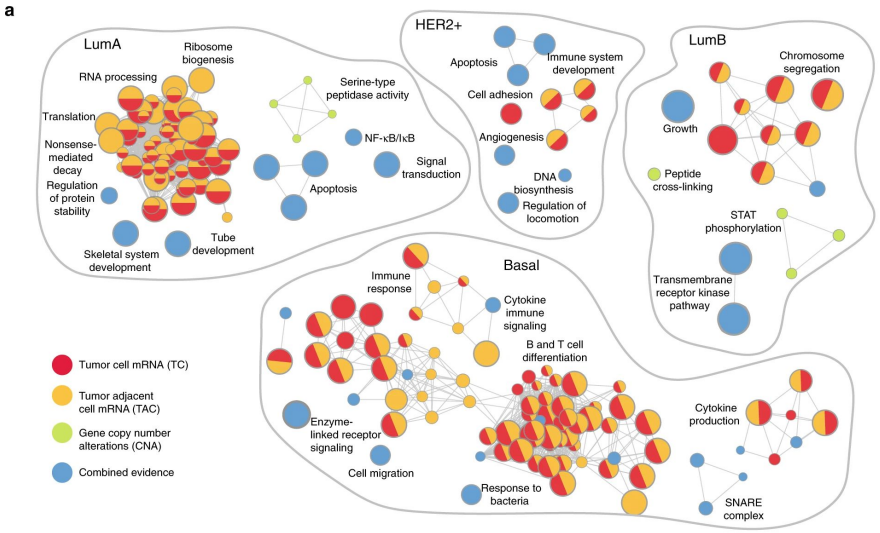
```
module load centos6/0.0.1-fasrc01
module load bedtools/2.17.0-fasrc01

#We can use bedtools to select a window size, then find the overlaps between the peaks and the annotated genes
bedtools window -w <window_size> -u -a <your_new_tss_bed_file> -b q4_diff_peaks_summits.bed > <your_output_bed>

#Count the number of genes
wc -l <your_output_bed>

#The same task might be also completed with the python package wrapping BEDTools (pybedtools), but you would need to install and import it
```

Then followed by BEDTools for checking if the peak regions are within +/- 2KB of the gene TSS

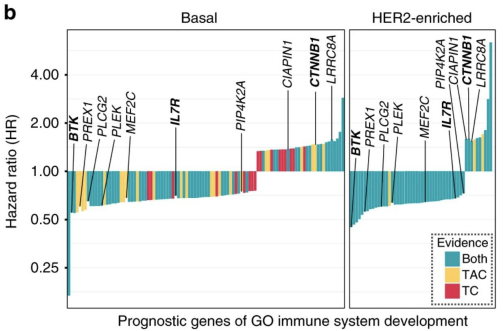
- Recommend BEDTools window

| Option | Description |
|---|---|
| **-abam** | BAM file A. Each BAM alignment in A is compared to B in search of overlaps. Use "stdin" if passing A with a UNIX pipe: For example: samtools view -b <BAM> \| bedtools window -abam stdin -b genes.bed |
| **-ubam** | Write uncompressed BAM output. The default is write compressed BAM output. |
| **-bed** | When using BAM input (-abam), write output as BED. The default is to write output in BAM when using -abam. For example: bedtools window -abam reads.bam -b genes.bed -bed |
| **-w** | Base pairs added upstream and downstream of each entry in A when searching for overlaps in B. *Default is 1000 bp.* |
| **-l** | Base pairs added upstream (left of) of each entry in A when searching for overlaps in B. *Allows one to create asymmetrical "windows". Default is 1000bp.* |
| **-r** | Base pairs added downstream (right of) of each entry in A when searching for overlaps in B. *Allows one to create asymmetrical "windows". Default is 1000bp.* |
| **-sw** | Define -l and -r based on strand. For example if used, -l 500 for a negative-stranded feature will add 500 bp downstream. *By default, this is disabled.* |
| **-sm** | Only report hits in B that overlap A on the same strand. *By default, overlaps are reported without respect to strand.* |
| **-Sm** | Only report hits in B that overlap A on the opposite strand. *By default, overlaps are reported without respect to strand.* |
| **-u** | Write original A entry once if any overlaps found in B. In other words, just report the fact at least one overlap was found in B. |
| **-c** | For each entry in A, report the number of hits in B while restricting to -w, -l, and -r. Reports 0 for A entries that have no overlap with B. |
| **-v** | Only report those entries in A that have *no overlaps* with B. |
| **-header** | Print the header from the A file prior to results. |

# Gene Ontology (GO) Analysis/ Pathway Enrichment Analysis




Enrichment maps of prognostic pathways and processes in different cell types.




Hazard ratios (HR) of prognostic genes related to immune system development in basal and HER2-enriched subtypes of breast cancer.

Some Examples of GO Analysis & Usage.

(Paczkowska et al. Nat Comm, 2020)

## Q13: For Graduate Students Only

Now overlap the putative AR target genes you get from above with up regulated genes in prostate cancers. Try to run GO (e.g. **DAVID**, you can try other ones too) analysis on 1) the **AR target genes from binding alone** and 2) the **AR target genes by overlapping AR binding with differential expression**. Are there enriched GO terms or pathways?

Hint: We have pre-computed the up-regulated genes by comparing a large number of prostate tumors with normal prostates, and the results are in /n/stat115/2021/HW3/up_regulated_genes_in_prostate_cancer.txt in Cannon.

AR target genes from binding **alone**:

The final genes from Q12

AR target genes by Overlapping AR binding with **differential expression**:

Overlapping Q12 gene names with up-regulated genes in prostate cancers

Read in the two tables, find (1) AR target genes alone and (2) Overlapping genes, then run DAVID.

Check ?read.table, ?intersect, ?write.table() in Rstudio Console (You can use python as well)

For **DAVID**: (Quick Live demo of DAVID in Lab)

For Identifier - Choose OFFICIAL_GENE_SYMBOL

# Question 14

Another way of getting the AR target genes is to consider the number of AR binding sites within **100KB of TSS,** but weight each binding site by an exponential decay of its distance to the gene TSS (i.e. peaks closer to TSS have higher weights). For this, we have calculated regulatory potential score for each refseq gene. Select the **top 1500 genes** with highest regulatory potential score, try to run GO analysis both with and without differential expression (overlap with the up-regulated genes), and see the enriched GO terms. **

Hints:

2) Basically this approach assumes that there are stronger AR targets (e.g. those genes with many AR binding sites within 100KB and have stronger differential expression) and weaker AR targets, instead of a **binary** Yes / No AR targets.

Read in the two tables

Find (1) top 1500 genes with highest regulatory potentials and (2) overlapping gene names with up-regulated genes in prostate cancer , then run DAVID.

For **DAVID**:

For Identifier - Choose OFFICIAL_GENE_SYMBOL

Check ?read.table, ?order, ?intersect and ?write.table in Rstudio Console (You can use python as well)

# Question 15  For Graduate Students Only

Comment on the AR targets you get from promoter binding (your code) and distance weighted binding. Which one gives you better function / pathway enrichment? Does considering differential expression help?

Include some screenshots of different GO results, and explain your reasonings.
Free-response here.

# Question 16

For what you did in Q12-15, Cistrome-GO (http://go.cistrome.org/) already provides a very simple solution. It performs functional enrichment analysis using a ChIP-seq peak file and an optional differential expression analysis file. Without differential expression, Cistrome-GO will perform GO analysis only on the regulatory potential. Now, use the differential peaks and upregulated genes to run Cistrome-GO and see the enriched biological functions or pathways.

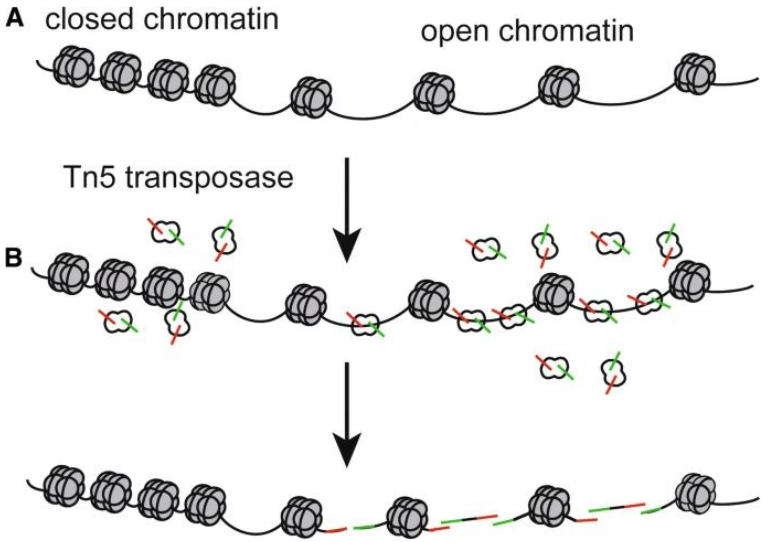
Live Demo of Cistrome-GO in Lab

Cistrome-GO (2019, *Nucleic Acids Research*):

1. Assign adjusted regulatory potentials score to each peak
2. Perform pathway enrichment analysis

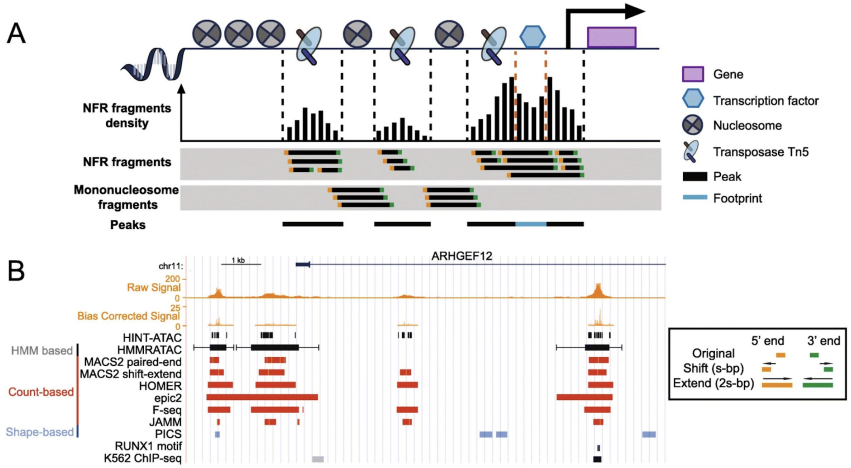
# ATAC-Seq
# (Part VI of HW3)

# Introduction of ATAC-Seq




Example of ATAC-Seq

( Sun et al., Hereditas, 2019)




( Yan et al., Genome Biology, 2020)

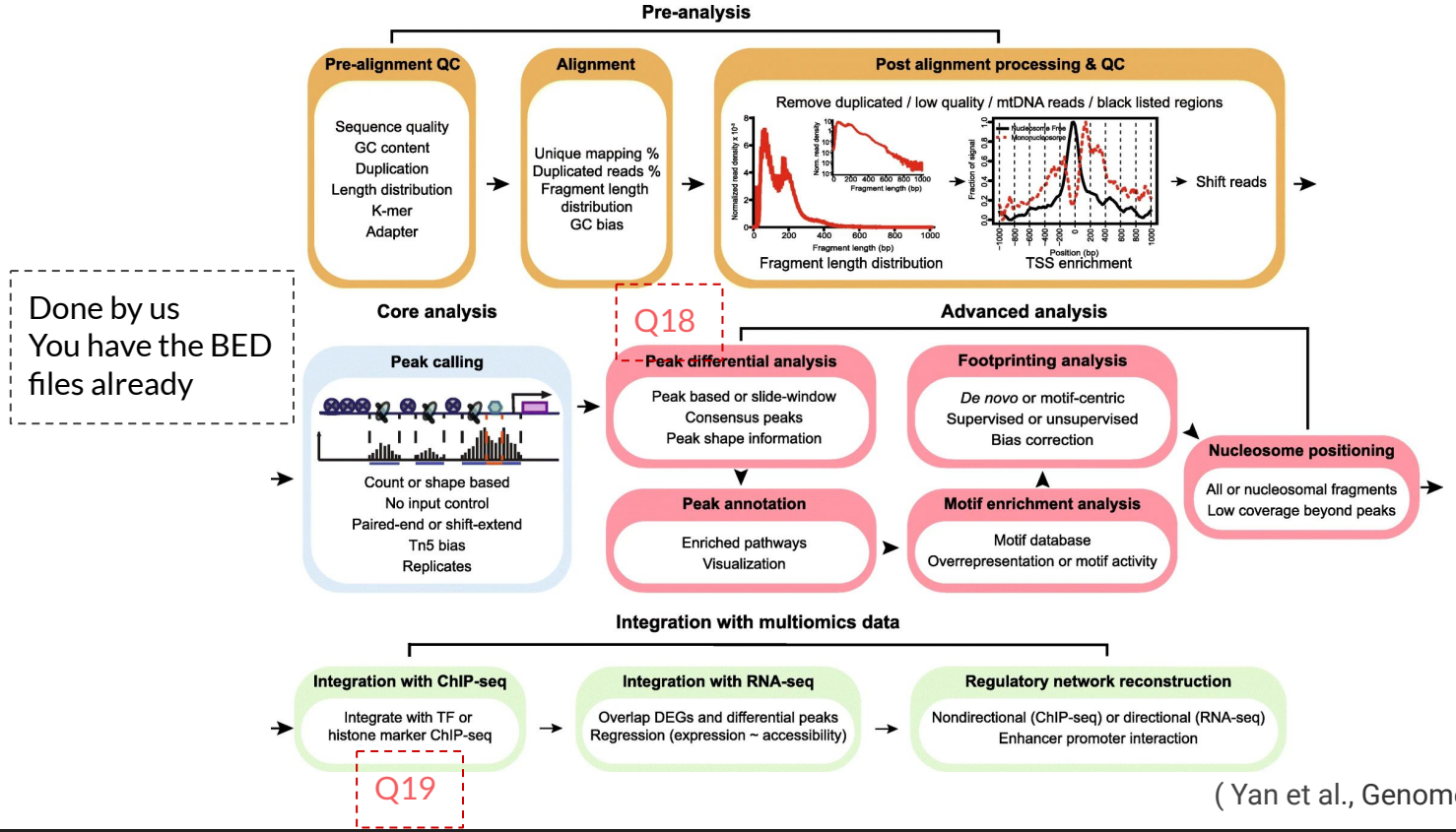
ATAC-Seq:

Measures **open chromatin** region.

Other chromatin accessibility measures:

DNASE-Seq, 4C, Hi-C, etc....

# Overall Pipeline of ATAC-Seq

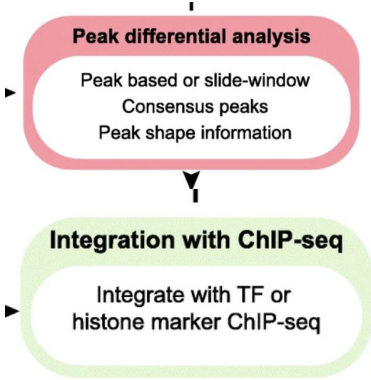



( Yan et al., Genome Biology, 2020)

# Question 18

We have found some published ATAC-seq data on prostate tumor and normal prostate tissues, and have done read mapping and the peak calls (MACS2) on each data separately. Use **BEDTools to find peaks that are unique in the tumor and not in the normal** (the 3rd approach). How many tumor-specific peaks (peaks in tumor but not in normal) do you get this way? Run this through **Cistrome Toolkit** and examine the results. What transcription factors are important in driving these prostate cancer-specific signals?

```
module load centos6/0.0.1-fasrc01
module load bedtools/2.17.0-fasrc01

your bedtool command | cut -f 1-3 > <your_out_put.bed>
```




- Take a look at BEDTools subtract documentation and their examples, then decide what should be your bedtools command (and what should be your -a file, what should be your -b file).
- Please include -A option.
- For all terminal commands, you can type `man <command_name>` for documentation. E.g. `man cut`
- Include a screen shot of the dynamic plot from Cistrome Toolkit.

# Question 19 For Graduate Students Only

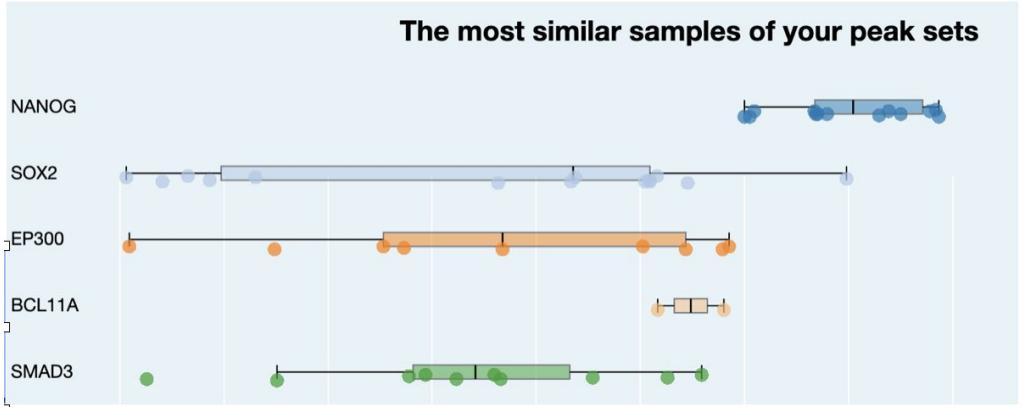
Now just take the top 10K ATAC-seq peaks (ranked by fold-change since these are all significant peaks already) in the **prostate tumor only** (not the differential peaks) and run **Cistrome Toolkit**. Compare the results with Q18. Can you comment on which approach gives you more meaning results?

Check ?read.table, ?order, and ?write.table in Rstudio Console for getting the top 10K peaks.
Include a screenshot of Cistrome Toolkit dynamic plot.

**Integration with ChIP-seq**

Integrate with TF or histone marker ChIP-seq

An example of output figure (Not HW data):



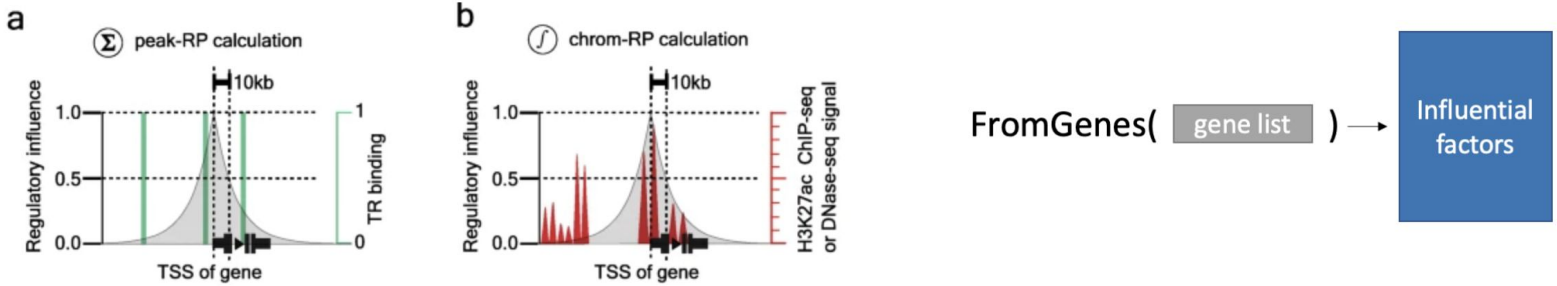
# Back to ChIP-Seq and Putative Target Genes (Q20 of HW3)

# LISA2

Documentation

- Infer regulatory TFs from public ChIP-Seq & DNASE-Seq datasets

Fig. 1




(Qin et al., Genome Biology, 2020)

# Question 20

Sometimes even without ATAC-seq, we can predict the driving transcription factors based on differentially expressed genes by using public ChIP-seq data. Even if the public TF ChIP-seq data was generated on different cells, they still provide some insights on the TF putative targets. Based on the differential gene expression list, Lisa first tries to build an epigenetic model using public ChIP-seq and chromatin accessibility profiles, then uses public ChIP-seq data to see which TF ChIP-seq fits the chromatin model the best. Now run the **up-regulated gene in prostate cancer** on Lisa, and see what transcription factors were predicted as putative drivers for these differential genes?

```
module load Anaconda/5.0.1-fasrc02

# use pre-installed lisa2 & acticate environment
source /n/stat115/2021/HW5/miniconda3/bin/activate
conda activate lisa2

lisa oneshot hg38 <your/up_regulated_gene_names_file/txt> -b 300 --seed=2556 --save_metadata >
</your/output/txt>
```

Then take a look at the output file, and provide a screenshot.
Does the TFs make sense?

# 2 Bonus Questions

1. Find a dataset with batch-effect and re-run HW2 Part I analysis
2. Rewrite all HW2 Part II's questions in sklearn.

# Q&A Time