1. Move to folder ATAC-seq, and create folders to store bigBed data files and peaks analyses files. Make sure the files are organized in a consistent way as done for ChIP-se

We start by: downloading ATAC-seq metadata and bigBed peak files from ENCODE for sigmoid colon and stomach samples. It filters for high-quality pseudoreplicated peaks aligned to the GRCh38 genome, downloads the relevant files, and checks their integrity using MD5 checksums to ensure nothing got corrupted. It sets you up with clean, trustworthy peak data for downstream analysis.

karim$ cd "/Users/karim/Desktop/MSc Omics Data Analysis/Courses/Epigenomics/Epigenomics Command/Epigenomics Hands On/epigenomics_uvic"

sudo docker run -v "$PWD":"$PWD" -w "$PWD" --rm -it dgarrimar/epigenomics_course

cd ATAC-seq

mkdir results raw_inputs genome_annotations

mkdir results/signal_peaks
mkdir raw_inputs/bigbed_data
mkdir raw_inputs/bed_data

2. Retrieve from a newly generated metadata file ATAC-seq peaks (bigBed narrow, pseudoreplicated peaks, assembly GRCh38) for stomach and sigmoid_colon for the same donor used in the previous sections. Hint: have a look at what we did here. Make sure your md5sum values coincide with the ones provided by ENCODE.

../bin/download.metadata.sh \
"https://www.encodeproject.org/metadata/?replicates.library.biosample.donor.uuid=d370683e-81e7-473f-8475-7716d027849b&status=released&status=submitted&status=in+progress&assay_title=ATAC-seq&biosample_ontology.term_name=sigmoid+colon&biosample_ontology.term_name=stomach&type=Experiment"

ls

grep -F "bigBed_narrowPeak" metadata.tsv | \
grep -F "pseudoreplicated_peaks" | \
grep -F "GRCh38" | \
awk 'BEGIN{FS=OFS="\t"}{print $1, $11}' | \
sort -k2,2 -k1,1r | \
sort -k2,2 -u > results/bigbed_peak_ids.tsv

```
cut -f1 results/bigbed_peak_ids.tsv | while read filename; do
  wget -P data/bigBed.files "https://www.encodeproject.org/files/$filename/@@download/$filename.bigBed"
done


for file_type in bigBed; do

  ../bin/selectRows.sh <(cut -f1 results/bigbed_peak_ids.tsv) metadata.tsv | cut -f1,46 > raw_inputs/"${file_type}"_data/md5sum.txt

  cat raw_inputs/"${file_type}"_data/md5sum.txt | \
  while read filename original_md5sum; do
    md5sum raw_inputs/"${file_type}"_data/"${filename}.${file_type}" | \
    awk -v filename="$filename" -v original_md5sum="$original_md5sum" 'BEGIN{FS=" "; OFS="\t"}{print filename, original_md5sum, $1}'
  done > tmp

  mv tmp raw_inputs/"${file_type}"_data/md5sum.txt

  awk '$2 != $3' raw_inputs/"${file_type}"_data/md5sum.txt

done
```

3. For each tissue, run an intersection analysis using BEDTools: report 1) the number of peaks that intersect promoter regions, 2) the number of peaks that fall outside gene coordinates (whole gene body, not just the promoter regions). Hint: have a look at what we did here and here.

The code below fetches ATAC-seq peak files (bigBed narrowPeak, pseudoreplicated, GRCh38 assembly) for stomach and sigmoid colon tissues from the ENCODE project for a specific donor. It downloads metadata, filters for relevant files, extracts unique bigBed file IDs, and downloads those files. Then it verifies data integrity by comparing the downloaded files' md5 checksums against the official ENCODE checksums, reporting any mismatches.

```
cut -f1 results/bigbed_peak_ids.tsv | while read filename; do
  bigBedToBed raw_inputs/bigbed_data/"${filename}.bigBed" raw_inputs/bigbed_data/"${filename}.bed"
done
```

```
mkdir -p genome_annotations
wget -P genome_annotations "https://public-docs.crg.es/rguigo/Data/bborsari/UVIC/epigenomics_course/gencode.v24.protein.coding.non.redundant.TSS.bed"



cut -f1,2 results/bigbed_peak_ids.tsv | while read filename tissue; do
   bedtools intersect -a raw_inputs/bigbed_data/"${filename}.bed" \
                -b genome_annotations/gencode.v24.protein.coding.non.redundant.TSS.bed -u \
   > results/signal_peaks/overlappedpeaks_promoter"${tissue}".bed
done

root@c86268081c42:/Users/karim/Desktop/MSc Omics Data Analysis/Courses/Epigenomics/Epigenomics Command/Epigenomics Hands On/epigenomics_uvic/ATAC-seq/results/signal_peaks# wc -l overlappedpeaks_promotersigmoid_colon.bed
47871 overlappedpeaks_promotersigmoid_colon.bed
root@c86268081c42:/Users/karim/Desktop/MSc Omics Data Analysis/Courses/Epigenomics/Epigenomics Command/Epigenomics Hands On/epigenomics_uvic/ATAC-seq/results/signal_peaks# wc -l overlappedpeaks_promoterstomach.bed
44749 overlappedpeaks_promoterstomach.bed
root@c86268081c42:/Users/karim/Desktop/MSc Omics Data Analysis/Courses/Epigenomics/Epigenomics Command/Epigenomics Hands On/epigenomics_uvic/ATAC-seq/results/signal_peaks#


cp "/Users/karim/Desktop/MSc Omics Data Analysis/Courses/Epigenomics/Epigenomics Command/Epigenomics Hands On/epigenomics_uvic/ChIP-seq/annotation/gencode.v24.protein.coding.gene.body.bed" \
  "/Users/karim/Desktop/MSc Omics Data Analysis/Courses/Epigenomics/Epigenomics Command/Epigenomics Hands On/epigenomics_uvic/ATAC-seq/genome_annotations/"



cut -f1,2 results/bigbed_peak_ids.tsv | while read filename tissue; do
   bedtools intersect -a raw_inputs/bigbed_data/"$filename".bed \
                -b genome_annotations/gencode.v24.protein.coding.gene.body.bed -v \
   > results/signal_peaks/peaks_outside_genes_"$tissue".txt
done
```

root@c86268081c42:/Users/karim/Desktop/MSc Omics Data Analysis/Courses/Epigenomics/Epigenomics Command/Epigenomics Hands On/epigenomics_uvic/ATAC-seq# wc -l results/signal_peaks/peaks_outside_genes_*.txt
  37035 results/signal_peaks/peaks_outside_genes_sigmoid_colon.txt
  34537 results/signal_peaks/peaks_outside_genes_stomach.txt
  71572 total

TASK5:
1: Create a folder regulatory_elements inside epigenomics_uvic. This will be the folder where you store all your subsequent results.

mkdir regulatory_elements

cd regulatory_elements

mkdir regions_bed
mkdir regions_bigbed
mkdir results_regulatory
mkdir results_regulatory/peak_insights

cp ChIP-seq/metadata.tsv regulatory_elements/ (do it from inside epigenomics_uvic)

2. Distal regulatory regions are usually found to be flanked by both H3K27ac and H3K4me1. From your starting catalogue of open regions in each tissue, select those that overlap peaks of H3K27ac AND H3K4me1 in the corresponding tissue. You will get a list of candidate distal regulatory elements for each tissue. How many are they?

This script identifies candidate distal regulatory elements by selecting open chromatin regions overlapping both H3K27ac and H3K4me1 histone modification peaks in stomach and sigmoid colon tissues. It downloads relevant bigBed files for each histone mark, verifies their md5 checksums, converts bigBed to BED format, then uses bedtools to intersect these BED files with previously identified ATAC-seq open regions outside genes. The final output counts the distal regulatory peaks overlapping both marks in each tissue, yielding 14,215 for sigmoid colon and 8,022 for stomach.

```
grep -F H3K27ac metadata.tsv | \
grep -F "bigBed_narrowPeak" | \
grep -F "pseudoreplicated_peaks" | \
grep -F "GRCh38" | \
awk 'BEGIN{FS=OFS="\t"}{print $1, $11}' | \
sort -k2,2 -k1,1r | \
sort -k2,2 -u > results_regulatory/H3K27ac_peak_ids.tsv
```

```
cut -f1 results_regulatory/H3K27ac_peak_ids.tsv | while read filename; do
   wget -P regions_bigbed "https://www.encodeproject.org/files/${filename}/@@download/${filename}.bigBed"
done
```

```
grep -F H3K4me1 metadata.tsv | \
grep -F "bigBed_narrowPeak" | \
grep -F "pseudoreplicated_peaks" | \
grep -F "GRCh38" | \
awk 'BEGIN{FS=OFS="\t"}{print $1, $11}' | \
sort -k2,2 -k1,1r | \
sort -k2,2 -u > results_regulatory/H3K4me1_peak_ids.tsv

cut -f1 results_regulatory/H3K4me1_peak_ids.tsv | while read filename; do
   wget -P regions_bigbed "https://www.encodeproject.org/files/${filename}/@@download/${filename}.bigBed"
done
```

```
cat  results_regulatory/H3K27ac_peak_ids.tsv  results_regulatory/H3K4me1_peak_ids.tsv  | cut -f1 | \
../bin/selectRows.sh - metadata.tsv | cut -f1,46 > regions_bigbed/combined_md5sum.txt
```

```
cat regions_bigbed/combined_md5sum.txt | while read filename original_md5sum; do
  md5sum regions_bigbed/"${filename}".bigBed | \
  awk -v f="$filename" -v o="$original_md5sum" 'BEGIN{FS=" "; OFS="\t"}{print f, o, $1}'
done > regions_bigbed/md5_comparison.txt

awk '$2 != $3' regions_bigbed/md5_comparison.txt
```

```
cut -f1 results_regulatory/H3K27ac_peak_ids.tsv | while read filename; do
   bigBedToBed regions_bigbed/"${filename}.bigBed" regions_bed/"${filename}.bed"
done

cut -f1 results_regulatory/H3K4me1_peak_ids.tsv | while read filename; do
   bigBedToBed regions_bigbed/"${filename}.bigBed" regions_bed/"${filename}.bed"
done
```

```
bedtools intersect -a ../ATAC-seq/results/signal_peaks/peaks_outside_genes_sigmoid_colon.txt -b regions_bed/ENCFF724ZOF.bed -u > results_regulatory/peak_insights/ENCFF724ZOF_overlaps.txt

bedtools intersect -a results_regulatory/peak_insights/ENCFF724ZOF_overlaps.txt -b regions_bed/ENCFF872UHN.bed -u > results_regulatory/peak_insights/distal_sigmoid_colon_peaks.txt

bedtools intersect -a ../ATAC-seq/results/signal_peaks/peaks_outside_genes_stomach.txt \
        -b regions_bed/ENCFF844XRN.bed -u \
        > results_regulatory/peak_insights/ENCFF844XRN_overlaps_stomach.txt

bedtools intersect -a results_regulatory/peak_insights/ENCFF844XRN_overlaps_stomach.txt \
        -b regions_bed/ENCFF977LBD.bed -u \
        > results_regulatory/peak_insights/distal_stomach_colon_peaks.txt

###go to peaks_insights
 wc -l distal_*.txt
  14215 distal_sigmoid_colon_peaks.txt
   8022 distal_stomach_colon_peaks.txt
  22237 total
```

3. Focus on regulatory elements that are located on chromosome 1 (hint: to parse a file based on the value of a specific column, have a look at what we did here), and generate a file regulatory.elements.starts.tsv that contains the name of the regulatory region (i.e. the name of the original ATAC-seq peak) and the start (5') coordinate of the region.

This script filters distal regulatory elements located specifically on chromosome 1 from previously generated peak files for sigmoid colon and stomach. It extracts the regulatory region names (from column 4) and their start coordinates (from column 2), then combines these into new two-column TSV files listing each region's name and start position. Temporary files used for extraction are deleted afterward.

```
grep '^chr1' distal_sigmoid_colon_peaks.txt | cut -f4 > tmp_col4_sigmoid.txt
```

```
grep '^chr1' distal_sigmoid_colon_peaks.txt | cut -f2 > tmp_col2_sigmoid.txt
paste tmp_col4_sigmoid.txt tmp_col2_sigmoid.txt > distal_sigmoid_custom_starts.tsv

grep '^chr1' distal_stomach_colon_peaks.txt | cut -f4 > tmp_col4_stomach.txt
grep '^chr1' distal_stomach_colon_peaks.txt | cut -f2 > tmp_col2_stomach.txt
paste tmp_col4_stomach.txt tmp_col2_stomach.txt > distal_stomach_custom_starts.tsv

rm tmp_col4_*.txt tmp_col2_*.txt
```

4. Focus on protein-coding genes located on chromosome 1. From the BED file of gene body coordinates that you generated here, prepare a tab-separated file called gene.starts.tsv which will store the name of the gene in the first column, and the start coordinate of the gene on the second column (REMEMBER: for genes located on the minus strand, the start coordinate will be at the 3'). Use the command below as a starting point:

```
###do the following from ChIP-seq---annotation and put it in our regulatory elements
cp ../ChIP-seq/annotation/gencode.v24.protein.coding.gene.body.bed .


awk 'BEGIN{FS=OFS="\t"} $1=="chr1" {pos=($6=="+") ? $2 : $3; print $4, pos}' gencode.v24.protein.coding.gene.body.bed > chr1_gene_tss_positions.tsv
```

5. Download or copy this python script inside the epigenomics_uvic/bin folder. Have a look at the help page of this script to understand how it works:
6. For each regulatory element contained in the file regulatory.elements.starts.tsv, retrieve the closest gene and the distance to the closest gene using the python script you created above. Use the command below as a starting point done > regulatory_ elements.genes.distances.sigmoid colon.tsv
_
7. Use R to compute the mean and the median of the distances stored in regulatoryElements.genes.distances.tsv.

the script is downloading the Python script get.distance.py into the epigenomics_uvic/bin folder and reviewing its help page to understand its usage for finding the closest gene and calculating the distance from a genomic coordinate based on a reference gene TSS file. Next, for each regulatory element listed in the file regulatory.elements.starts.tsv, which includes the element name and start position, the script is run to retrieve the nearest gene and the distance to that gene. This is accomplished by iterating through each regulatory element's start coordinate and providing it as input to the Python script along with the gene TSS reference file, saving the results separately for stomach and sigmoid colon tissues. Finally, R is used to load the resulting distance data files and compute summary statistics—specifically

the mean and median distances—thereby quantifying how far regulatory elements are from their closest genes. ***The python script does not work, even though I tried to update it or fix it, so I could not get the means eventually***

```
python ../bin/get.distance.py --input chr1_gene_tss_positions.tsv --start 5000
ENSG00000186092.4  69090  64090

while read element start; do
   python ../../../bin/get.distance.py --input distal_stomach_custom_starts.tsv --start $start
done < distal_stomach_custom_starts.tsv > stomach_gene_distance.tsv

while read element start; do
   python ../../../bin/get.distance.py --input distal_sigmoid_custom_starts.tsv --start $start
done < distal_sigmoid_custom_starts.tsv > sigmoid_gene_distance.tsv

Rscript -e 'data <read.table("stomach_gene_distance.tsv", header=FALSE);
print(mean(data$V3)); print(median(data$V3))'

Rscript -e 'data <read.table("sigmoid_gene_distance.tsv", header=FALSE);
print(mean(data$V3)); print(median(data$V3))'
```