4. EN-TEx ATAC-seq data: downstream analyses

To start this practice, the first step is to run the following docker container:

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

Next, after accessing the epigeonimics_uvic folder created during the previous practical at class, we create a new folder called ATAC-seq. After that, we move into it:

```
cd epigenomics_uvic
mkdir ATAC-seq
cd ATAC-seq
```

Within this folder, we create two more folders called analyses which is going to contain peaks analyses and data for bigBed Data. It is important to make sure that our files are organized in a consistent way as we did for ChIP-seq analysis.

```
mkdir analyses
mkdir data
```

The following step consists of downloading peak calling, for which we must first create the following folders.

```
mkdir data/bigBed.files
mkdir data/bigWig.files
```

The next step requires the creation of a new metadata file which is accessed via the link in the following code.

```
../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&biosample_ontology.term_name=stomach&biosample_ontology.term_name=sigmoid+colon&assay_title=ATAC-seq&assay_slims=DNA+accessibility&type=Experiment

head -1 metadata.tsv | awk 'BEGIN{FS=OFS="\t"}{for (i=1;i<=NF;i++){print $i, i}}'
```

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. To this end, we have to first download the corresponding files (for peak calling and fold-change signals).

The next step consists of obtain their corresponding IDs. In this instance, we save the files of interest and call them bigBed.peaks.ids.txt. Although we are in the ATAC-seq directory we can indicate that we want to save our new txt file in the analyses directory.

```
grep -F "bigBed_narrowPeak" metadata.tsv|\
grep -F "pseudoreplicated_peaks" |\
grep -F "GRCh38" |\
awk 'BEGIN{FS=OFS="\t"}{print $1, $11, $23}' |\
sort -k2,2 -k1,1r |\
sort -k2,2 -u > analyses/bigBed.peaks.ids.txt

cd analyses
```

Then if we go to the analyses directory we can take a look to the new .txt generated which contain our IDs of interest: ENCFF287UHP for sigmoid_colon and ENCFF762IFP for stomach. The following step is to download the files from the link via the IDs saved in the file we have just created.

```
cd ..
cut -f1 analyses/bigBed.peaks.ids.txt |\
while read filename; do
    wget -P data/bigBed.files "https://www.encodeproject.org/files/$filename/@@download/$file-
name.bigBed"
done
```

Next, it is interesting to look if the files have been correctly generated (which have been located in the bigBed.files file which is in data). We can see that there are two new files named ENCFF287UHP.bigBed, and ENCFF762IFP.bigBed.

The following step is to tetrieve the original MD5 hash from the metadata:

```
../bin/selectRows.sh <(cut -f1 analyses/"bigBed".*.ids.txt) metadata.tsv | cut -f1,46 > data/md5sum.txt
```

We see that a file called md5sum.txt is created in the data folder. When we open it we see the following code:

ENCFF762IFP	f6a97407b6ba4697108e74451fb3eaf4
ENCFF287UHP	46f2ae76779da5be7de09b63d5c2ceb9

Then, we have to compute Compute MD5 hash on the downloaded files:

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

Now, the output which is in the md5sum.txt belongs to the following codes:

ENCFF762IFP	f6a97407b6ba4697108e74451fb3eaf4	f6a97407b6ba4697108e74451fb3eaf4
ENCFF287UHP	46f2ae76779da5be7de09b63d5c2ceb9	46f2ae76779da5be7de09b63d5c2ceb9

Finally, it is interesting to make sure that there are no files for which original and computed MD5 hashes differ. As we are not getting any output, we can assume that it is correct.

```
awk '$2!=$3' md5sum.txt
```

The next thing we must do is for each tissue, run an intersection analysis using BEDTools. To do so, we need to create a new directory we must create a new directory to save the annotation we downloaded from the following link. So, the first thing is to obtain the gencode.v24.protein.coding.non.redundant.TSS.bed and check that it has the promoter regions.

```
mkdir annotation
wget -P annotation https://public-docs.crg.es/rguigo/Data/bborsari/UVIC/epigenomics_course/gen-
code.v24.protein.coding.non.redundant.TSS.bed
```

1. **The first task is to compute the number of peaks that intersect promoter regions. To this end we need to create a new folder in the data directory which is going to be called bed.files. Moreover, we are generating another file which is called peaks.analyses located in the analyses directory to save the output.**

```
mkdir data/bed.files
mkdir analyses/peaks.analyses
```

We can then compute the intersection through bedtools intersect. In order to do this, we first transform the .bigBed files into .bed files.

```
cut -f1 analyses/bigBed.peaks.ids.txt |\
while read filename; do
    bigBedToBed data/bigBed.files/"$filename".bigBed data/bed.files/"$filename".bed
done
```

Next, as in all cases, we explore the files that have been generated in the bed.files file. There are two files which are named ENCFF287UHP.bed and ENCFF762IFP.bed.

```
head ENCFF287UHP.bed
```

For instance, this .bed file returns the following code. Therefore, the next step is to perform the intersection. This is done between the obtained annotation with the non-redundant TSS and the .bed files containing the peaks.

```
chr1    778339  779193  Peak_175893     36      .       2.20588 3.66321 1.70835 781
chr1    778339  779193  Peak_203412     32      .       2.10748 3.27805 1.40370 60
chr1    778339  779193  Peak_2485       1000    .       26.44889        265.42572       261.44012       425
chr1    817296  818240  Peak_23845      396     .       7.14286 39.65236        36.84049        208
chr1    817296  818240  Peak_54490      127     .       3.85948 12.74064        10.30683        778
chr1    817296  818240  Peak_67130      95      .       3.32518 9.53628 7.19598 553
chr1    818645  819307  Peak_57080      119     .       3.77937 11.91532        9.50266 404
chr1    818645  819307  Peak_61510      107     .       3.59712 10.75067        8.37177 137
chr1    826914  828166  Peak_33159      254     .       4.43365 25.44272        22.78304        377
chr1    826914  828166  Peak_6539       1000    .       13.19213        163.74141       160.25230       716
```

```
cut -f-2 analyses/bigBed.peaks.ids.txt |\
while read filename tissue; do
    bedtools intersect -wa -a data/bed.files/"$filename".bed -b annotation/gencode.v24.protein.coding.non.redundant.TSS.bed |\
     sort -u > analyses/peaks.promoters."$tissue".ATAC.bed
done
```

Once this code has been applied, we redirect to the analysis folder to see which files have been generated. So we see: peaks.promoters.sigmoid_colon.ATAC.bed and peaks.promoters.stomach.ATAC.bed. When we explore them we find:

```
cd analyses
```

```
head peaks.promoters.sigmoid_colon.ATAC.bed
```

```
chr1    100037869       100038981       Peak_27878      321     .       4.54433 32.19514        29.45567
67
chr1    100037869       100038981       Peak_3151       1000    .       15.08016        240.54111       236.68378
91
chr1    100037869       100038981       Peak_37434      215     .       3.73506 21.51265        18.90843
37
chr1    100037869       100038981       Peak_39666      198     .       3.60451 19.86863        17.29079
00
chr1    100132473       100133456       Peak_49540      146     .       3.49259 14.62506        12.14781
70
chr1    100132473       100133456       Peak_5313       1000    .       15.08818        186.48776       182.89174
74
chr1    100132473       100133456       Peak_53466      131     .       3.29564 13.14501        10.70213
96
chr1    1001849 1002073 Peak_30448      285     .       3.71640 28.51370        25.81558        115
chr1    100249509       100250300       Peak_1270       1000    .       21.73774        323.83047       319.51874
05
chr1    100249509       100250300       Peak_57105      119     .       3.13901 11.91260        9.50007
77
```

With all this, we can establish how many genes are found in each of the files. So, we apply wc -l for all files with extension .bed that are in the analyses folder. All in all, we get 47871 peaks in the file belonging to sigmoid colon and 44749 in the file pertaining to stomach.

```
wc -l *.bed
```

```
47871 peaks.promoters.sigmoid_colon.ATAC.bed
44749 peaks.promoters.stomach.ATAC.bed
92620 total
```

2. **In this second part of the exercise, we have to compute the number of peaks that fall outside gene coordinates (whole gene body, not just the promoter regions). In other words, we need the outer peaks.**

Again, the first step is to obtain the appropriate annotation. In this case, we want to get gencode.v24.protein.coding.gene.body.bed. We are going to extract it directly from the encode page (from the ATAC-seq directory but the resulting file will be saved in the annotation folder).

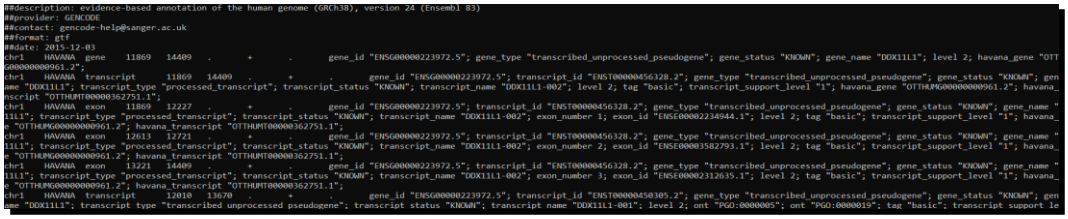
```
wget -P annotation https://www.encodeproject.org/files/gencode.v24.primary_assembly.annotation/@@download/gencode.v24.primary_assembly.annotation.gtf.gz
```

We see that inside the annotation folder we have the file gencode.v24.primary_assembly.annotation.gtf.gz. The next step is to unzip this file for this purpose:

```
gunzip annotation/gencode.v24.primary_assembly.annotation.gtf.gz
```

We then examine the resulting file:

```
less annotation/gencode.v24.primary_assembly.annotation.gtf
```



As we can clearly see, it is a .gtf file and therefore we must convert it to .bed.

```
awk '$3=="gene"' annotation/gencode.v24.primary_assembly.annotation.gtf |\
grep -F "protein_coding" |\
cut -d ";" -f1 |\
awk 'BEGIN{OFS="\t"}{print $1, $4, $5, $10, 0, $7, $10}' |\
sed 's/\"//g' |\
awk 'BEGIN{FS=OFS="\t"}$1!="chrM"{$2=($2-1); print $0}' > annotation/gencode.v24.protein.coding.gene.body.bed
```

After applying this code, we can see how the file gencode.v24.protein.coding.gene.body.bed has been generated in the annotation folder.

```
cut -f-2 analyses/bigBed.peaks.ids.txt |while read filename tissue; do bedtools intersect -b annotation/gencode.v24.protein.coding.gene.body.bed -a data/bed.files/"$filename".bed -v > analyses/outerpeaks."$tissue".bed; done

cd analyses

wc -l outer*.bed
```

Hence, we find 37035 peaks outside the gene coordinates in the sigmoid colon and 34357 corresponding to the stomach (outerpeaks.sigmoid_colon.bed and outerpeaks.stomach.bed).

```
37035 outerpeaks.sigmoid_colon.bed
34537 outerpeaks.stomach.bed
71572 total
```

5. Distal regulatory activity

1. **The first step of this second exercise is to create a regulatory_elements folder inside epigenomics_uvic. This will be the folder where all subsequent results will be stored.**

```
mkdir regulatory_elements
cd regulatory_elements
```

2. **Distal regulatory regions are usually found to be flanked by both H3K27ac and H3K4me1. From our 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. The following steps are followed to find out how many there are. First, we have to create two folders in regulatory_elements with H3K27acpeaks and H3K4me1peaks.**

```
mkdir H3K27acpeaks
mkdir H3K4me1peaks
```

Next, we need to create bed files with H3K27ac and H3K4me1 for each tissue using the metadata.tsv located in the Chip-seq directory previously used in the class practical (we are accessing to it through its path). In the first instance, we are going to store the output in the file bigBed.peaksH3K27ac.ids.txt located in H3K27acpeaks:

```
grep -F H3K27ac ../ChIP-seq/metadata.tsv | grep -F "bigBed_narrowPeak" | grep -F "pseudoreplicated_peaks" |grep -F "GRCh38" |awk 'BEGIN{FS=OFS="\t"}{print $1, $11, $23}' |sort -k2,2 -k1,1r |sort -k2,2 -u > H3K27acpeaks/bigBed.peaksH3K27ac.ids.txt
head H3K27acpeaks/bigBed.peaksH3K27ac.ids.txt
```

```
ENCFF872UHN     sigmoid_colon   H3K27ac-human
ENCFF977LBD     stomach H3K27ac-human
```

```
cut -f1 H3K27acpeaks/bigBed.peaksH3K27ac.ids.txt |\
while read filename; do wget -P H3K27acpeaks "https://www.encodeproject.org/files/$filename/@@download/$filename.bigBed"
done
```

We proceed to do the same approach but now with H3K4me1. In this case, the output is saved in bigBed.peaksH3K4me1.ids.txt at H3K4me1peaks:

```
grep -F H3K4me1 ../ChIP-seq/metadata.tsv |
grep -F "bigBed_narrowPeak" |
grep -F "pseudoreplicated_peaks" |
grep -F "GRCh38" |
awk 'BEGIN{FS=OFS="\t"}{print $1, $11, $23}' |
sort -k2,2 -k1,1r |
sort -k2,2 -u > H3K4me1peaks/bigBed.peaksH3K4me1.ids.txt

head H3K4me1peaks/bigBed.peaksH3K4me1.ids.txt
```

```
ENCFF724ZOF     sigmoid_colon   H3K4me1-human
ENCFF844XRN     stomach H3K4me1-human
```

```
cut -f1 H3K4me1peaks/bigBed.peaksH3K4me1.ids.txt |\
while read filename; do
        wget -P H3K4me1peaks "https://www.encodeproject.org/files/$filename/@@download/$filename.bigBed"
done
```

Prior to performing the intersection, we need to convert the bigBed files to bed files. For this purpose, we use the function bigBedtoBed.

```
cut -f1 H3K27acpeaks/bigBed.peaksH3K27ac.ids.txt |\
while read filename; do bigBedToBed H3K27acpeaks/"$filename".bigBed H3K27acpeaks/"$filename".bed
done
```

```
root@6b753cc24132:~/epigenomics_uvic/regulatory_elements/H3K27acpeaks# head ENCFF872UHN.bed
chr1    777817  779498  Peak_11032      860     .       15.81056        86.05346        83.52045        720
chr1    816703  817505  Peak_18296      324     .       8.83107 32.48223        30.36750        610
chr1    817917  820175  Peak_17448      356     .       9.37175 35.63359        33.48695        1283
chr1    826635  827928  Peak_6449       1000    .       26.64176        158.91512       155.91089       796
chr1    904299  905189  Peak_26537      164     .       5.84944 16.48707        14.57866        309
chr1    910289  910576  Peak_71728      47      .       2.92472 4.72662 3.12944 126
chr1    923982  924326  Peak_37712      97      .       4.26720 9.75193 7.98111 184
chr1    939957  942164  Peak_22981      211     .       6.84545 21.17233        19.19260        812
chr1    943638  944373  Peak_37219      98      .       4.32542 9.89267 8.11765 411
chr1    958797  959335  Peak_16239      413     .       8.34583 41.36764        39.16735        321
```

Next, we perform the same procedure but with the .bigBed from the H3K4me1.

```
cut -f1 H3K4me1peaks/bigBed.peaksH3K4me1.ids.txt |\
while read filename; do bigBedToBed H3K4me1peaks/"$filename".bigBed H3K4me1peaks/"$filename".bed
done
```

```
root@6b753cc24132:~/epigenomics_uvic/regulatory_elements/H3K4me1peaks# head ENCFF844XRN.bed
chr1    825765  826161  Peak_23295      184     .       3.64679 9.18052 6.41664 105
chr1    904000  904494  Peak_46295      127     .       3.16074 6.81763 4.40744 266
chr1    904801  905859  Peak_20059      199     .       3.32591 9.77814 6.93866 472
chr1    909985  911437  Peak_23309      184     .       3.64679 9.18052 6.41664 256
chr1    916167  917334  Peak_19084      201     .       3.76831 9.80847 6.93866 961
chr1    919944  920264  Peak_46341      127     .       3.16074 6.81763 4.40744 98
chr1    920358  921662  Peak_8357       275     .       3.95504 12.89295        9.60725 612
chr1    923103  925817  Peak_1409       447     .       5.46948 19.89991        15.63772        517
chr1    932726  933123  Peak_59552      108     .       2.95734 6.03531 3.76173 176
chr1    933388  933891  Peak_13164      233     .       4.01133 11.10580        8.04792 255
```

The next step is to apply bedtools intersect of H3K4me1 with those peaks that are outside the gene coordinates of each tissue (in the files peakoutside_sigmoidcolon.bed and peakoutisde_stomach.bed).

```
bedtools intersect -a ../ATAC-Seq/peaksoutside_sigmoidcolon.bed -b H3K4me1peaks/ENCFF724ZOF.bed -u > H3K4me1peaks/common_outside_sigmoidcolon.bed
```

```
root@6b753cc24132:~/epigenomics_uvic/regulatory_elements/H3K4me1peaks# head common_outside_sigmoidcolon.bed
chr1    817296  818240  Peak_23845      396     .       7.14286 39.65236        36.84049        208
chr1    817296  818240  Peak_54490      127     .       3.85948 12.74064        10.30683        778
chr1    817296  818240  Peak_67130      95      .       3.32518 9.53628 7.19598 553
chr1    858673  859020  Peak_18124      569     .       10.41931        56.99270        54.04943        171
chr1    904265  904921  Peak_22173      435     .       7.26132 43.57509        40.73002        498
chr1    904265  904921  Peak_39088      202     .       4.68198 20.29614        17.71163        205
chr1    910645  911472  Peak_56282      121     .       3.85617 12.19396        9.77472 686
chr1    910645  911472  Peak_68095      92      .       3.31992 9.29615 6.96278 111
chr1    921111  921389  Peak_21707      448     .       6.06811 44.80150        41.94651        132
chr1    923679  924127  Peak_21162      463     .       6.25197 46.38569        43.51812        148
```

```
bedtools intersect -a ../ATAC-seq/peaksoutside_stomach.bed -b H3K4me1peaks/ENCFF844XRN.bed -u > H3K4me1peaks/common_outside_stomach.bed
```

```
root@6b753cc24132:~/epigenomics_uvic/regulatory_elements/H3K4me1peaks# head common_outside_stomach.bed
chr1    904635  904862  Peak_37530      163     .       4.20770 16.38258        13.76219        114
chr1    910571  910892  Peak_71563      80      .       3.01887 8.00946 5.66739 174
chr1    921152  921333  Peak_56686      102     .       3.04748 10.26586        7.82583 90
chr1    923712  923969  Peak_73931      76      .       2.74390 7.65829 5.33100 103
chr1    1059260 1059450 Peak_125716     47      .       2.17539 4.76179 2.63927 106
chr1    1059551 1059992 Peak_17692      382     .       5.81081 38.27313        35.31115        123
chr1    1059551 1059992 Peak_237301     30      .       1.85440 3.07862 1.25371 378
chr1    1067682 1068269 Peak_25860      249     .       3.88170 24.92832        22.14262        271
chr1    1067682 1068269 Peak_68978      82      .       2.39135 8.22404 5.86863 515
chr1    1068516 1069461 Peak_19319      346     .       4.48689 34.61676        31.69713        267
```

Next, we apply the same code but in the case of the peaks belonging to H3K27acpeaks.

```
bedtools intersect -a H3K4me1peaks/common_outside_sigmoidcolon.bed -b H3K27acpeaks/ENCFF872UHN.bed -u > common_sigmoid_colon.bed
```

```
root@6b753cc24132:~/epigenomics_uvic/regulatory_elements# head common_sigmoid_colon.bed
chr1    817296  818240  Peak_23845      396     .       7.14286 39.65236        36.84049        208
chr1    817296  818240  Peak_54490      127     .       3.85948 12.74064        10.30683        778
chr1    817296  818240  Peak_67130      95      .       3.32518 9.53628 7.19598 553
chr1    904265  904921  Peak_22173      435     .       7.26132 43.57509        40.73002        498
chr1    904265  904921  Peak_39088      202     .       4.68198 20.29614        17.71163        205
chr1    923679  924127  Peak_21162      463     .       6.25197 46.38569        43.51812        148
chr1    1122088 1122426 Peak_20098      497     .       7.93796 49.74874        46.85592        212
chr1    1157380 1158638 Peak_233916     27      .       1.69724 2.73909 0.98229 1124
chr1    1157380 1158638 Peak_24251      387     .       5.33258 38.79213        35.98783        676
chr1    1157380 1158638 Peak_62286      105     .       2.81069 10.52377        8.15081 971
```

```
bedtools intersect -a H3K4me1peaks/common_outside_stomach.bed -b H3K27acpeaks/ENCFF977LBD.bed -u > common_stomach.bed
```

```
root@6b753cc24132:~/epigenomics_uvic/regulatory_elements# head common_stomach.bed
chr1    1067682 1068269 Peak_25860      249     .       3.88170 24.92832        22.14262        271
chr1    1067682 1068269 Peak_68978      82      .       2.39135 8.22404 5.86863 515
chr1    1068516 1069461 Peak_19319      346     .       4.48689 34.61676        31.69713        267
chr1    1068516 1069461 Peak_24518      264     .       3.63030 26.47291        23.66292        825
chr1    1079493 1080378 Peak_24039      270     .       5.22928 27.02500        24.20646        525
chr1    1079493 1080378 Peak_25265      256     .       5.01567 25.61829        22.82155        195
chr1    1079493 1080378 Peak_38063      160     .       4.00490 16.07549        13.46184        768
chr1    1124797 1125018 Peak_32494      192     .       3.54911 19.25146        16.56763        106
chr1    1125097 1125536 Peak_106662     55      .       2.09450 5.51351 3.32675 326
chr1    1125097 1125536 Peak_127062     46      .       1.98758 4.68771 2.57039 81
```

Lastly, we count the lines to determine the number of candidate distal regulatory elements for each tissue (for all those .bed files).

wc -l *.bed

```
14215 common_sigmoid_colon.bed
 8022 common_stomach.bed
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.**

We then select the peaks found in chr1 for both tissues and save them separately in the files sigmoid-colon.regulatory.elements.start.tsv and stomach.regulatory.elements.start.tsv.

cut -f2 ../ATAC-seq/analyses/bigBed.peaks.ids.txt |\

> while read tissue; do grep -w chr1 common$tissue.bed | awk 'BEGIN{FS=OFS="\t"}$1=="chr1"{print $4, $2}' > $tissue.regulatory.elements.starts.tsv

> done

```
root@6b753cc24132:~/epigenomics_uvic/regulatory_elements# head sigmoid_colon.regulatory.elements.starts.tsv
Peak_23845      817296
Peak_54490      817296
Peak_67130      817296
Peak_22173      904265
Peak_39088      904265
Peak_21162      923679
Peak_20098      1122088
Peak_233916     1157380
Peak_24251      1157380
Peak_62286      1157380
```

```
root@6b753cc24132:~/epigenomics_uvic/regulatory_elements# head stomach.regulatory.elements.starts.tsv
Peak_25860      1067682
Peak_68978      1067682
Peak_19319      1068516
Peak_24518      1068516
Peak_24039      1079493
Peak_25265      1079493
Peak_38063      1079493
Peak_32494      1124797
Peak_106662     1125097
Peak_127062     1125097
```

wc -l sigmoid_colon.regulatory.elements.starts.tsv stomach.regulatory.elements.starts.tsv

```
root@6b753cc24132:~/epigenomics_uvic/regulatory_elements# wc -l sigmoid_colon.regulatory.elements.starts
.tsv stomach.regulatory.elements.starts.tsv
 1521 sigmoid_colon.regulatory.elements.starts.tsv
  987 stomach.regulatory.elements.starts.tsv
 2508 total
```

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. For this purpose, we will use gencode.v24.protein.coding.gene.body.bed found in the annotation file in ATAC-seq and focus on protein-coding genes found on chromosome 1.**

grep -w chr1 ../ATAC-seq/annotation/gencode.v24.protein.coding.gene.body.bed |

awk 'BEGIN{FS=OFS="\t"}{if ($6=="+"){start=$2} else {start=$3}; print $4, start}' > gene.starts.tsv

```
root@6b753cc24132:~/epigenomics_uvic/regulatory_elements# head gene.starts.tsv
ENSG00000186092.4       69090
ENSG00000279928.1       182392
ENSG00000279457.3       200322
ENSG00000278566.1       451678
ENSG00000273547.1       686654
ENSG00000187634.10      924879
ENSG00000188976.10      959309
ENSG00000187961.13      960586
ENSG00000187583.10      966496
ENSG00000187642.9       982093
```

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:

First, we must copy all the code provided in the link. To do this, we use nano and create a file called get.distance.py.

cd bin
nano get.distance.py

Below is how the lines of code would look like in the nano editor.

```
#************
# LIBRARIES *
#************

import sys
from optparse import OptionParser


#*****************
# OPTION PARSING *
#*****************

parser = OptionParser()
parser.add_option("-i", "--input", dest="input")
parser.add_option("-s", "--start", dest="start")
options, args = parser.parse_args()

open_input = open(options.input)
enhancer_start = int(options.start)


#********
# BEGIN *
#********

x=1000000 # set maximum distance to 1 Mb
selectedGene="" # initialize the gene as empty
selectedGeneStart=0 # initialize the start coordinate of the gene as empty

for line in open_input.readlines(): # for each line in the input file
        gene, y = line.strip().split('\t') # split the line into two columns based on a tab
        # define a variable called position that correspond to the integer of the start of the gene
        # compute the absolute value of the difference between position and enhancer_start

        # if this absolute value is lower than x
                # this value will now be your current x
                # save gene as selectedGene
                # save position as selectedGeneStart

print "\t".join([selectedGene, str(selectedGeneStart), str(x)])
```

The next step, once we are already inside the regulatory_elements directory, we can reopen the file with nano and add the following lines.

nano ../bin/get.distance.py

```
root@0b753cc24132:~/epigenomics_uvic/regulatory_elements# cat ../bin/get.distance.py
#************
# LIBRARIES *
#************

import sys
from optparse import OptionParser


#*****************
# OPTION PARSING *
#*****************

parser = OptionParser()
parser.add_option("-i", "--input", dest="input")
parser.add_option("-s", "--start", dest="start")
options, args = parser.parse_args()

open_input = open(options.input)
enhancer_start = int(options.start)


#********
# BEGIN *
#********

x=1000000 # set maximum distance to 1 Mb
selectedGene="" # initialize the gene as empty
selectedGeneStart=0 # initialize the start coordinate of the gene as empty

for line in open_input.readlines(): # for each line in the input file
        gene, y = line.strip().split('\t') # split the line into two columns based on a tab
        position = int(y) # define a variable called position that correspond to the integer of the start of the gene
        absolute_value = position - enhancer_start # compute the absolute value of the difference between position and enhancer_start
        absolute_value = abs(absolute_value)
        if absolute_value < x: # if this absolute value is lower than x
                x= absolute_value # this value will now be your current x
                selectedGene = gene # save gene as selectedGene
                selectedGeneStart = position # save position as selectedGeneStart

print "\t".join([selectedGene, str(selectedGeneStart), str(x)])
```

Then, to make sure that it has been done correctly, we apply the following code and check if the result is correct.

python ../bin/get.distance.py --input gene.starts.tsv --start 980000

```
root@0b753cc24132:~/epigenomics_uvic/regulatory_elements# python ../bin/get.distance.py --input gene.starts.tsv --start 980000
ENSG00000187642.9       982093  2093
```

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:

cut -f2 ../ATAC-seq/analyses/bigBed.peaks.ids.txt |\

while read tissue; do cat $tissue.regulatory.elements.starts.tsv| while read element start; do python ../bin/get.distance.py --input gene.starts.tsv --start $start done > $tissue.regulatory.elements.genes.distances.tsv
done

```
root@6b753cc24132:~/epigenomics_uvic/regulatory_elements# cat sigmoid_colon.regulatory.elements.genes.d
stances.tsv
ENSG00000171163.15      248859144       4074
```

```
root@6b753cc24132:~/epigenomics_uvic/regulatory_elements# cat stomach.regulatory.elements.genes.distance
s.tsv
ENSG00000171163.15      248859144       3771
```

7. **Use R to compute the mean and the median of the distances stored in regulatoryElements.genes.distances.tsv. As we can see, it is very important to put the argument na.rm=TRUE because otherwise the result will be NA. So, the following figures summarize the results that have been obtained.**

```
> Sigmoid_csv <- read.csv("sigmoid_colon.regulatory.elements.genes.distances.tsv", header=F, sep="")
> distances_sigmoid_colon <- as.vector(unlist(Sigmoid_csv[3]))
> mean_sigmoidcolon <- mean(distances_sigmoid_colon)
> mean_sigmoidcolon
[1] NA
> mean_sigmoidcolon <- mean(distances_sigmoid_colon, na.rm=TRUE)
> mean_sigmoidcolon
[1] 73026.44
> median_sigmoidcolon <- median(distances_sigmoid_colon, na.rm=TRUE)
> median_sigmoidcolon
[1] 35768
```

```
> Stomach_csv <- read.csv("stomach.regulatory.elements.genes.distances.tsv", header=F, sep="")
> distances_stomach <- as.vector(unlist(Stomach_csv[3]))
> mean_stomach <- mean(distances_stomach, na.rm=TRUE)
> mean_stomach
[1] 45227.05
> median_stomach <- median(distances_stomach, na.rm=TRUE)
> median_stomach
[1] 27735
```