**Loops and Their Pile-up Analysis Pipeline**

The intended analysis of this pipeline is the pile-up (aggregate) analysis of the loops whose anchors overlap with peaks of particular histone marks or TF1,2. The input for pipeline is a normalized .cool matrix file.

1. **Call loops using Mustache**3**:**

Loops can be called using the following command. In the command below, -f is input .cool file, -ch is the list of chromosomes, -r is the resolution, and -o is the output file. If you don't specify the chromosome (-ch) for a .[m]cool mustache will run on all chromosomes and output loop anchors (coordinates) in a .tsv file specified by -o. For more information on arguments please refer to the [Mustache documentation](https://github.com/ay-lab/mustache?tab=readme-ov-file#mustache----).

path\_to\_mustache/mustache/mustache/mustache.py -f cool\_file\_path -ch chr\_list -r res -st 0.7 -pt 0.5 -p 4 -o out.tsv

1. **Identify loops whose anchors overlap with histone or TF peaks:**

In this step, we identify loops whose anchors overlap with peaks of particular histone marks or TFs. These selected loops will be used in the next step for pile-up analysis. The following Python script can identify loops whose left anchor overlaps with bed\_1 and right anchor overlaps with bed\_2. It outputs a .bedpe file containing identified loops. Please modify the input arguments (lines 6-9) as required before running.

1. **import** numpy as np
2. **import** pandas as pd
3. **import** pybedtools

6. loops\_file\_path **=** "../output/loops\_check/loops.tsv"                   # path to loops.tsv
7. bed\_1\_file\_path **=** "../input/bed\_files/DE\_Ctrl\_PRDM1.0.7rpm.bed"       # Left histone/TF mark
8. bed\_2\_file\_path **=** "../input/bed\_files/DE\_Ctrl\_H2AK119Ub1.all.bed"     # Right histone/TF mark
9. out\_file\_path **=** "../output/check.bedpe" # path to output file
11. # intersection
13. loops\_df**=**pd.read\_csv(loops\_file\_path, sep**=**"\t")
14. loops\_df["loop\_number"]**=**np.arange(loops\_df.shape[0])
16. loops\_left\_df**=**loops\_df[["BIN1\_CHR","BIN1\_START","BIN1\_END","loop\_number"]]
17. loops\_right\_df**=**loops\_df[["BIN2\_CHROMOSOME","BIN2\_START","BIN2\_END","loop\_number"]]
19. loops\_left\_bed**=**pybedtools.BedTool.from\_dataframe(loops\_left\_df)
20. loops\_right\_bed**=**pybedtools.BedTool.from\_dataframe(loops\_right\_df)
22. bed\_1 **=** pybedtools.BedTool(bed\_1\_file\_path)
23. bed\_2 **=** pybedtools.BedTool(bed\_2\_file\_path)
25. loops\_left\_bed\_1\_intersect**=**loops\_left\_bed.intersect(bed\_1, u**=**True).to\_dataframe()
26. loops\_left\_bed\_1\_intersect**=**loops\_left\_bed\_1\_intersect.rename(columns**=**{"chrom":"BIN1\_CHR", "start":"BIN1\_START", "end":"BIN1\_END", "name":"loop\_number"})
28. loops\_right\_bed\_2\_intersect**=**loops\_right\_bed.intersect(bed\_2, u**=**True).to\_dataframe()
29. loops\_right\_bed\_2\_intersect**=**loops\_right\_bed\_2\_intersect.rename(columns**=**{"chrom":"BIN2\_CHROMOSOME", "start":"BIN2\_START", "end":"BIN2\_END", "name":"loop\_number"})
31. loops\_bed\_1\_on\_left**=**loops\_df.merge(loops\_left\_bed\_1\_intersect, how**=**"inner", on**=**["BIN1\_CHR", "BIN1\_START", "BIN1\_END", "loop\_number"])
33. loops\_bed\_2\_on\_right **=** loops\_df.merge(loops\_right\_bed\_2\_intersect, how**=**"inner", on**=**["BIN2\_CHROMOSOME", "BIN2\_START", "BIN2\_END", "loop\_number"])
35. loops\_bed\_1\_on\_left\_bed\_2\_on\_right **=** loops\_bed\_1\_on\_left.merge(loops\_bed\_2\_on\_right, how**=**"inner")
37. loops\_bed\_1\_on\_left\_bed\_2\_on\_right.to\_csv(out\_file\_path, sep**=**"\t", header**=**None, index**=**False)
38. **Aggregate (pile-up) analysis:**

Aggregate analysis is carried out using coolpuppy package4. The script to perform pile-up analysis can be downloaded from [here](https://github.com/spg5958/micro_c_analysis_Iwafuchi_lab-github_repo/blob/main/scripts/loop_aggregate_analysis/3_pileup_analysis.py). It takes the loop file (in bedpe format) generated in the previous step and performs off-diagonal pile-up analysis using coolpuppy, producing a pile-up analysis plot. Please modify the input arguments (lines 13-18) at the beginning of the script as needed and run the script. After a successful run, you should obtain a plot of the pile-up analysis.

**References:**

1. Hsieh, T.-H. S. *et al.* Resolving the 3D Landscape of Transcription-Linked Mammalian Chromatin Folding. *Mol. Cell* **78**, 539-553.e8 (2020).

2. Hsieh, T.-H. S. *et al.* Enhancer–promoter interactions and transcription are largely maintained upon acute loss of CTCF, cohesin, WAPL or YY1. *Nat. Genet.* **54**, 1919–1932 (2022).

3. Mustache. https://github.com/ay-lab/mustache.

4. coolpuppy. https://github.com/open2c/coolpuppy/.