**WORKSHOP 1: Analysis of genome-wide transcription factor-binding sites using ChIP-seq data**

**TASK1: Create an environment using mamba, and name the environment “workshop1”**

**Address: 10.139.1.132 Username: binf6\_03 Password: binf-GiantDingo76**

mamba create -n workshop1 python=3.7

mamba activate workshop1

mamba install -c bioconda fastqc seqtk bowtie2 samtools macs2 idr

**TASK2: PIPELINE**

A computer screen shot of a black screen

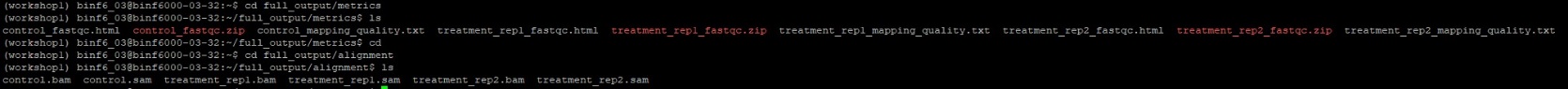
Description automatically generated

A screen shot of a computer

Description automatically generated

A black background with white lines

Description automatically generated



A computer screen with many lights

Description automatically generated with medium confidence

A screenshot of a computer program

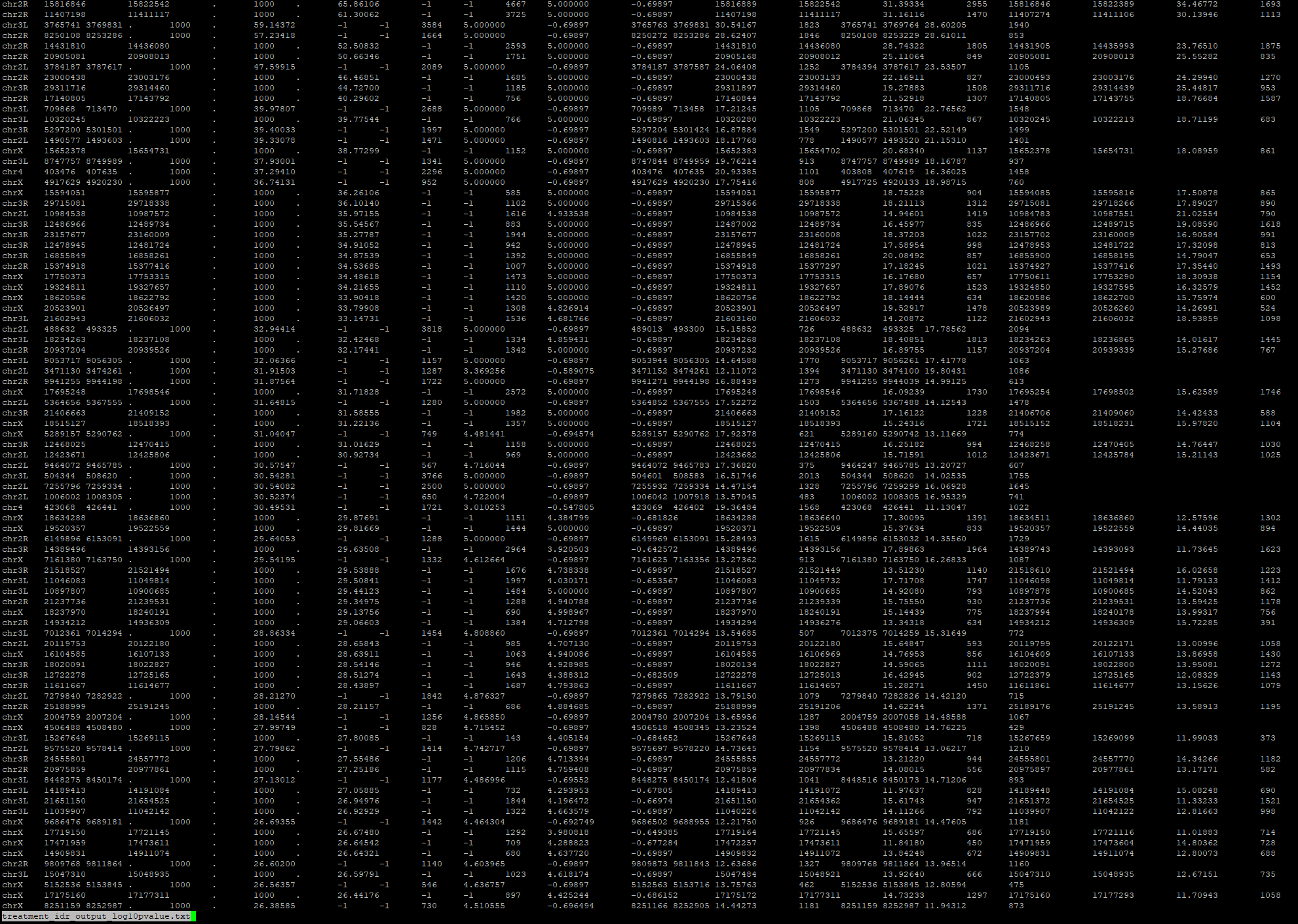
Description automatically generated

A screenshot of a computer program

Description automatically generated

A computer screen with text

Description automatically generated

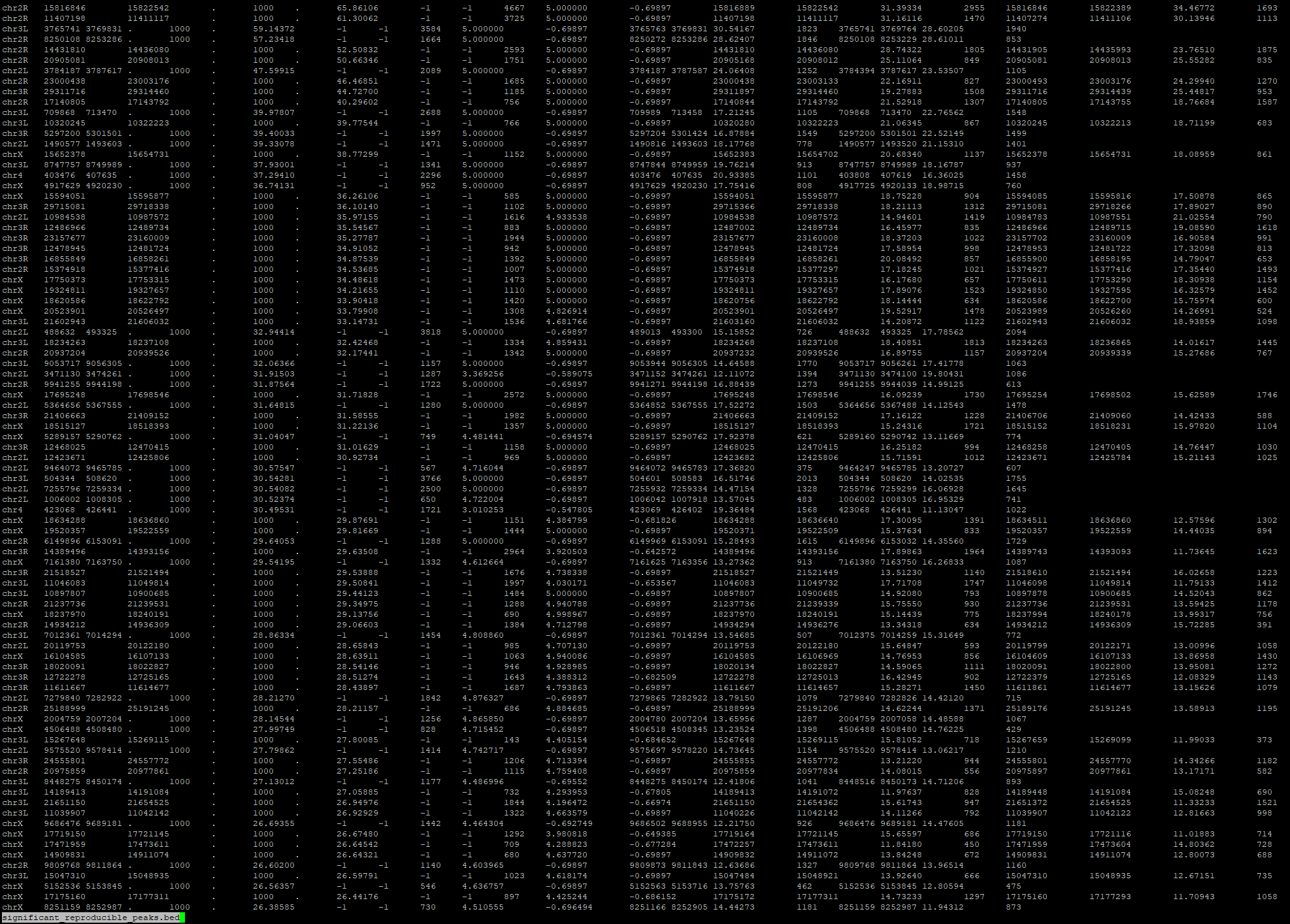
****

**A screenshot of a computer

Description automatically generated**

**TASK3: Identify a gene bound by the transcription factor CTCF**

less significant\_reproducible\_peaks.bed.



Given these locations a genome annotation file recording gene locations (dm6\_tss.bed), and a getclosestgene.py script, I can identify a gene which has a CTCF binding site in its promoter region.

$ python binfpy/getclosestgene.py home/binf6\_03/significant\_reproducible\_peaks.bed dm6\_tss.bed

This will output a bed file tss\_gene.bed detailing:

<CTCF bind chrom> <CTCF bind start> <CTCF bind end> <Gene name> <distance between CTCF and gene> <strand>

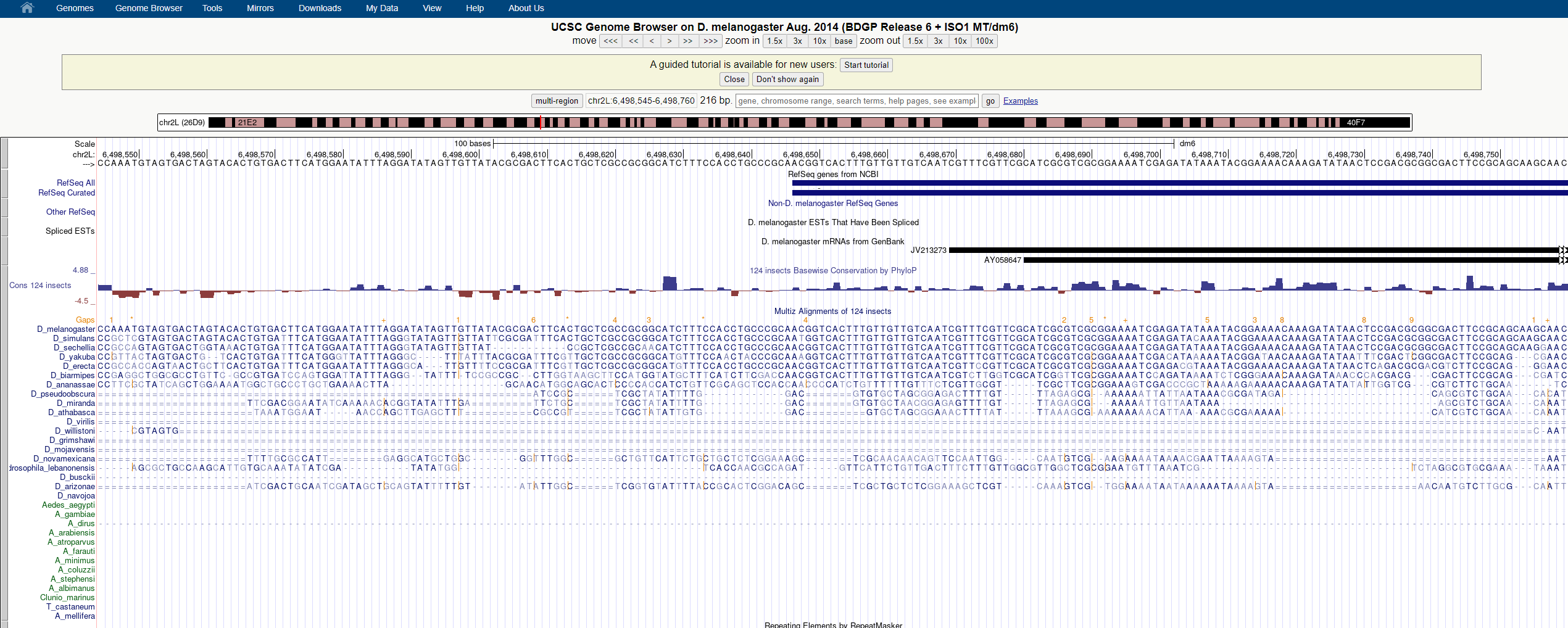
less tss\_gene.bed

A screenshot of a computer program

Description automatically generated

chr2L is the chromosome where the binding site is located. "chr2L" refers to the left arm of the second chromosome in Drosophila melanogaster. 6498467 is the start position of the CTCF binding site on the chromosome. BED file coordinates are 0-based, meaning that the first base of the chromosome is considered position 0. Thus, this binding site starts at the 6,498,467th base of chr2L. 6498838 is the end position of the CTCF binding site on the chromosome. In BED format, the end position is exclusive, meaning the actual binding site extends up to but does not include this position. Therefore, the binding site spans from base 6,498,467 to base 6,498,837, making it 371 bases long. NM\_001298748.1\_up\_1\_chr2L\_6498646\_f is a unique identifier for the binding site or the peak. It includes the gene name with which the site is associated, in this case, "NM\_001298748.1", which could be a gene identifier in a specific database. The additional details (up\_1\_chr2L\_6498646\_f) provide context about the binding site's location, such as it being upstream of the gene, its chromosome, a specific base position, and the direction ("f" for forward strand). The “0” represents the score of the peak or binding site, which can indicate the strength or confidence in the site's identification. A score of "0" might suggest a default value in this context. The "+" means that the binding site is on the forward strand, which has implications for the directionality of any genes or regulatory elements associated with this site.

Discovering a CTCF binding site located on the left arm of chromosome 2L, at the precise coordinates of 6,498,467 to 6,498,837, unveils a fascinating glimpse into the intricate regulatory networks within Drosophila melanogaster. This sequence lies in proximity to the gene tagged as NM\_001298748.1. Given CTCF's renowned role as an architectural protein, shaping the 3D organization of chromatin and dictating the rhythm of gene expression, this association is more than mere coincidence. It suggests a targeted regulatory influence, where CTCF could be modulating the expression of NM\_001298748.1, thereby impacting fundamental biological processes from development and cell differentiation to the safeguarding of chromosomal architecture. This discovery not only highlights the complexity and precision of genetic regulation in D. melanogaster but also opens up avenues for exploring how such regulatory mechanisms contribute to the organism's biology and evolution.



#!/bin/bash

FULL\_OUTPUT\_DIR="full\_output"

# Step 1: Quality Control

mkdir -p ${FULL\_OUTPUT\_DIR}/metrics

fastqc data/\*.fastq.gz -o ${FULL\_OUTPUT\_DIR}/metrics/

# Step 2: Alignment with Bowtie2

mkdir -p ${FULL\_OUTPUT\_DIR}/alignment.

bowtie2 -x reference/bowtie2\_index/genome \

-U data/control.fastq.gz \ # Specifies the path to the FASTQ file for the control sample.

> ${FULL\_OUTPUT\_DIR}/alignment/control.sam

bowtie2 -x reference/bowtie2\_index/genome \ # Specifies the path to the Bowtie2 index.

-U data/treatment\_rep1.fastq.gz \

> ${FULL\_OUTPUT\_DIR}/alignment/treatment\_rep1.sam

bowtie2 -x reference/bowtie2\_index/genome \

-U data/treatment\_rep2.fastq.gz \ # Specifies the path to the FASTQ file for treatment replicate 2.

> ${FULL\_OUTPUT\_DIR}/alignment/treatment\_rep2.sam

# Step 3: SAM to BAM Conversion and Mapping Quality Check

samtools view -bS ${FULL\_OUTPUT\_DIR}/alignment/control.sam \

> ${FULL\_OUTPUT\_DIR}/alignment/control.bam

samtools view -bS ${FULL\_OUTPUT\_DIR}/alignment/treatment\_rep1.sam \

> ${FULL\_OUTPUT\_DIR}/alignment/treatment\_rep1.bam

samtools view -bS ${FULL\_OUTPUT\_DIR}/alignment/treatment\_rep2.sam \

> ${FULL\_OUTPUT\_DIR}/alignment/treatment\_rep2.bam

samtools flagstat ${FULL\_OUTPUT\_DIR}/alignment/control.bam \

> ${FULL\_OUTPUT\_DIR}/metrics/control\_mapping\_quality.txt

samtools flagstat ${FULL\_OUTPUT\_DIR}/alignment/treatment\_rep1.bam \

> ${FULL\_OUTPUT\_DIR}/metrics/treatment\_rep1\_mapping\_quality.txt

samtools flagstat ${FULL\_OUTPUT\_DIR}/alignment/treatment\_rep2.bam \

> ${FULL\_OUTPUT\_DIR}/metrics/treatment\_rep2\_mapping\_quality.txt

# Step 4: Peak Calling with MACS2

mkdir -p ${FULL\_OUTPUT\_DIR}/peaks

macs2 callpeak -t ${FULL\_OUTPUT\_DIR}/alignment/treatment\_rep1.bam \ # Specify the treatment replicate 1 BAM file as the target.

-c ${FULL\_OUTPUT\_DIR}/alignment/control.bam \

-f BAM -g dm -n treatment\_rep1 \

--outdir ${FULL\_OUTPUT\_DIR}/peaks -q 0.05

--call-summits --nomodel --extsize 147

macs2 callpeak -t ${FULL\_OUTPUT\_DIR}/alignment/treatment\_rep2.bam \

-c ${FULL\_OUTPUT\_DIR}/alignment/control.bam \

-f BAM -g dm -n treatment\_rep2 \

--outdir ${FULL\_OUTPUT\_DIR}/peaks -q 0.05 \

--call-summits --nomodel --extsize 147

# Step 5: Reproducibility Assessment with IDR

mkdir -p ${FULL\_OUTPUT\_DIR}/idr # Create a directory for storing idr results.

idr --samples ${FULL\_OUTPUT\_DIR}/peaks/treatment\_rep1\_peaks.narrowPeak \

${FULL\_OUTPUT\_DIR}/peaks/treatment\_rep2\_peaks.narrowPeak \

--output-file ${FULL\_OUTPUT\_DIR}/idr/treatment\_idr\_output.txt \

--log-output-file ${FULL\_OUTPUT\_DIR}/idr/treatment\_idr.log

awk 'BEGIN {OFS="\t"} {$12 = -log($12)/log(10); print}' ${FULL\_OUTPUT\_DIR}/idr/treatment\_idr\_output.txt > ${FULL\_OUTPUT\_DIR}/idr/treatment\_idr\_output\_log10p>

# Step 6: Filtering Peaks Based on IDR Output

mkdir -p ${FULL\_OUTPUT\_DIR}/final\_peaks

awk '$12 < 0.05' ${FULL\_OUTPUT\_DIR}/idr/treatment\_idr\_output.txt \

> ${FULL\_OUTPUT\_DIR}/final\_peaks/significant\_reproducible\_peaks.bed

SUBSET

#!/bin/bash

# Step 0: Create subsets of the original fastq files for testing purposes to speed up the process.

echo "Subsetting data for testing..."

mkdir -p sub\_data # Create a directory for the subsetted data.

# Use seqtk to sample 1000 reads from the original fastq.gz files, then gzip the output for FastQC compatibility.

seqtk sample -s100 data/control.fastq.gz 10000 | gzip > sub\_data/sub\_control.fastq.gz

seqtk sample -s100 data/treatment\_rep1.fastq.gz 10000 | gzip > sub\_data/sub\_treatment\_rep1.fastq.gz

seqtk sample -s100 data/treatment\_rep2.fastq.gz 10000 | gzip > sub\_data/sub\_treatment\_rep2.fastq.gz

# Step 1: Run FastQC for quality control on the subsetted fastq.gz files.

echo "Running FastQC for quality control..."

mkdir -p output/metrics # Create a directory for FastQC reports.

fastqc sub\_data/\*.fastq.gz -o output/metrics/ # Run FastQC on all subsetted fastq.gz files and output the reports to the metrics directory.

# Step 2: Align reads to the reference genome with Bowtie2.

mkdir -p output/alignment # Create a directory for the alignment output.

# Align each subsetted fastq.gz file to the reference genome using Bowtie2, outputting SAM files.

bowtie2 -x reference/bowtie2\_index/genome -U sub\_data/sub\_control.fastq.gz > output/alignment/sub\_control.sam

bowtie2 -x reference/bowtie2\_index/genome -U sub\_data/sub\_treatment\_rep1.fastq.gz > output/alignment/sub\_treatment\_rep1.sam

bowtie2 -x reference/bowtie2\_index/genome -U sub\_data/sub\_treatment\_rep2.fastq.gz > output/alignment/sub\_treatment\_rep2.sam

# Step 3: Convert SAM files to BAM format and check mapping quality.

echo "Converting SAM to BAM and checking mapping quality..."

# Convert SAM to BAM using samtools for each alignment file.

samtools view -bS output/alignment/sub\_control.sam > output/alignment/sub\_control.bam

samtools view -bS output/alignment/sub\_treatment\_rep1.sam > output/alignment/sub\_treatment\_rep1.bam

samtools view -bS output/alignment/sub\_treatment\_rep2.sam > output/alignment/sub\_treatment\_rep2.bam

# Generate mapping quality reports for each BAM file using samtools flagstat.

samtools flagstat output/alignment/sub\_control.bam > output/metrics/sub\_control\_mapping\_quality.txt

samtools flagstat output/alignment/sub\_treatment\_rep1.bam > output/metrics/sub\_treatment\_rep1\_mapping\_quality.txt

samtools flagstat output/alignment/sub\_treatment\_rep2.bam > output/metrics/sub\_treatment\_rep2\_mapping\_quality.txt

# Step 4: Call peaks using MACS2 with a fixed extension size.

mkdir -p output/peaks # Create a directory for peak calling output.

# Use MACS2 to call peaks on each treatment BAM file against the control, specifying parameters like genome size, q-value cutoff, etc.

macs2 callpeak -t output/alignment/sub\_treatment\_rep1.bam -c output/alignment/sub\_control.bam -f BAM -g dm -n sub\_treatment\_rep1 --outdir output/peaks -q 0.05 --call-summits --nomodel --extsize 147

macs2 callpeak -t output/alignment/sub\_treatment\_rep2.bam -c output/alignment/sub\_control.bam -f BAM -g dm -n sub\_treatment\_rep2 --outdir output/peaks -q 0.05 --call-summits --nomodel --extsize 147

# Step 5: Use IDR (Irreproducible Discovery Rate) to assess the reproducibility between biological replicates' peak sets.

echo "Assessing reproducibility with IDR..."

idr --samples output/peaks/sub\_treatment\_rep1\_peaks.narrowPeak \

output/peaks/sub\_treatment\_rep2\_peaks.narrowPeak \

--output-file output/idr/sub\_treatment\_idr\_output.txt \

--plot \

--log-output-file output/idr/sub\_treatment\_idr.log

# Step 6: Filter for significant, reproducible peaks based on IDR output.

echo "Filtering significant, reproducible peaks based on IDR output..."

# Use awk to filter the IDR output for entries with a column 12 value (adjusted p-value) less than 0.05, indicating significant reproducibility.

awk '$12 < 0.05' output/idr/sub\_treatment\_idr\_output.txt > output/final\_peaks/significant\_reproducible\_peaks.bed

echo "Pipeline execution for subsetted data completed."