Supplemental Information for 'Pioneer factor GAF cooperates with PBAP and NURF to regulate transcription'

Julius Judd* Fabiana M. Duarte† John T. Lis*‡

*Department of Molecular Biology and Genetics, Cornell University, Ithaca, New York 14835, USA
†Broad Institute of MIT and Harvard, Cambridge, MA 02142, USA
‡Correspondance to johnlis@cornell.edu

Table of Contents

The code in this file is also available at https://github.com/JAJ256/GAF

Supplementary Table 1. Oligonucleotides used for generating RNAi templates

Name	Sequence
LACZ-fwd	GAATTAATACGACTCACTATAGGGAGAGATATCCTGCTGATGAAGC
LACZ-rev	GAATTAATACGACTCACTATAGGGAGAGCAGGAGCTCGTTATCGC
GAF-fwd	GAATTAATACGACTCACTATAGGGATGGTTATGTTGGCTGGCGTCAA
GAF-rev	GAATTAATACGACTCACTATAGGGATCTTTACGCGTGGTTTGCGT
BAP170-fwd	GAATTAATACGACTCACTATAGGGTGGACGGAATAGAGCTACCTGG
BAP170-rev	GAATTAATACGACTCACTATAGGGTCAATGGAGCGAGAGGTGG
NURF301-fwd	GAATTAATACGACTCACTATAGGGATGTTGAATACTGGTTGACATAGTTC
NURF301-rev	GAATTAATACGACTCACTATAGGGAGTGCTAATCCGGCATGATA

Supplementary Table 2. PRO-seq alignment metrics

RNAi	Rep.	Raw Reads	% Adapter	% rRNA	% PCR Dups	Uniq. Non-Dup	Scale Factor
BAP170	1	39,244,087	6.63%	22.05%	0.01%	15,379,401	0.626245
BAP170	2	26,110,434	8.35%	20.05%	0.01%	9,304,431	1.000000
GAF	1	48,692,340	6.43%	19.10%	0.02%	20,159,679	0.364060
GAF	2	29,022,776	5.40%	20.27%	0.01%	11,329,566	0.650773
LACZ	1	40,319,830	7.44%	22.28%	0.01%	16,451,927	0.533495
LACZ	2	35,740,698	5.82%	20.88%	0.01%	14,594,540	0.579465
NURF301+BAP170	1	45,591,479	8.56%	16.97%	0.01%	14,763,333	0.522120
NURF301+BAP170	2	38,224,454	7.09%	16.19%	0.01%	12,152,986	0.556803
NURF301	1	61,102,403	7.90%	10.55%	0.01%	17,854,770	0.559650
NURF301	2	34,094,767	7.09%	10.40%	0.01%	9,103,431	0.964054

Supplementary Table 3. ATAC-seq alignment metrics

RNAi	Rep.	Raw Reads	Uniq. Aligned Reads
BAP170	1	38,500,034	30,231,402
BAP170	2	14,483,253	11,502,911
GAF	1	21,962,543	17,108,608
GAF	2	22,832,804	17,912,219
LACZ	1	23,978,897	19,095,473
LACZ	2	18,977,303	15,141,294
NURF301+BAP170	1	24,629,540	19,568,654
NURF301+BAP170	2	17,109,780	13,816,496
NURF301	1	22,298,270	17,243,075
NURF301	2	18,414,735	14,401,208

Supplementary Table 4. 3'RNA-seq alignment metrics

RNAi	Rep.	Raw Reads	% Adapter	% PCR Dups	Uniq. Non-Dup	ERCC Reads	Scale Factor
BAP170	1	9,897,071	4.75%	68.53%	3,114,663	113,089	0.696478
BAP170	2	12,083,788	4.17%	71.77%	3,411,719	122,112	0.645014
GAF	1	10,572,796	5.76%	71.71%	2,991,223	104,509	0.753658
GAF	2	16,673,438	4.67%	74.68%	4,222,261	131,954	0.596905
LACZ	1	14,553,663	5.35%	72.75%	3,966,094	105,724	0.744996
LACZ	2	13,133,097	9.67%	73.64%	3,461,387	78,764	1.000000
NURF301+BAP170	1	12,153,411	5.52%	74.60%	3,086,672	111,297	0.707692
NURF301+BAP170	2	22,904,765	9.37%	76.98%	5,271,654	160,525	0.490665
NURF301	1	12,878,694	5.98%	73.75%	3,380,145	117,876	0.668194
NURF301	2	10,521,245	5.22%	70.39%	3,115,527	101,947	0.772598

Supplementary Table 5. CUT&RUN alignment metrics

Target	Raw Reads	% Adapter	Uniq. Aligned Reads
GAF	18,499,020	2.34%	12,097,380
NURF301	17,234,208	2.13%	10,713,798

Supplementary Table 6. External data sources

Description	PMID	GEO	SRA	Remapped?
M1BP ChIP-seq	23708796	GSE49842	SRP028808	No
BEAF-32 ChIP-seq	24486021	GSE52962	SRP033490	Yes
GAF ChIP-seq	25815464	GSE40646	SRP015432	Yes
M1BP PRO-seq	27492368	GSE77607	SRP069335	No

Supplementary Equation 1. Spike-In normalization strategy

$$Signal[i]_{normalized} = Signal[i]_{raw} \cdot \frac{min\{SpikeIn_i \ ... \ SpikeIn_n\}}{SpikeIn_i}$$

Where:

$Signal[i]_{normalized}$ = Normalized signal for sample i

$Signal[i]_{raw}$ = Raw counts for sample i

$min\{SpikeIn_i \ ... \ SpikeIn_n\}$ = Minimum number spike-in reads mapped across all samples

$SpikeIn_i$ = Number of spike-in reads mapped for sample i

Supplementary Code 1. PRO-seq alignment pipeline

This pipeline can be found here:

http://github.com/jaj256/PROseq_alignment.sh

Analysis in this paper was performed using commit 55a08db

```
#!/bin/bash

############################################################
# Tue Mar  5 14:04:36 EST 2019                             #
# This is a pipeline script for handling paired end        #
# PRO-seq data with UMIs on both ends of the read.         #
# Run this script in a directory that has one folder       #
# named "fastq" which contains the data.                   #
# Fastq files must have identical names other than         #
# ending in _R1.fastq and _R2.fastq.                       #
############################################################

## Parameters
THREADS=50 # Threads to use for multithreaded applications
UMI_LEN=6  # Length of UMI in basepairs

## UMI Flags (set to Y or N as appropriate)
FIVEP_UMI="Y" # Is there a UMI on the 5' end of the read?
THREEP_UMI="Y" # Is there a UMI on the 3' end of the read?

## Adaptor sequences to clip. Default = Tru-Seq small RNA
ADAPTOR_1="TGGAATTCTCGGGTGCCAAGGAACTCCAGTCAC"
ADAPTOR_2="GATCGTCGGACTGTAGAACTCTGAACGTGTAGATCTCGGTGGTCGCCGTATCATT"

## Genomes. Fill in paths.
GENOME_EXP="/home/jaj256/genome/dm6/dm6Hsp70AaOnly"
GENOME_SPIKE="/home/jaj256/genome/dm6hg38/dm6hg38" ## USE REPEAT MASKED VERSION!!
SPIKE_PREFIX="hg38" ## This is the prefix you've used on your spike in chromosomes
RDNA="/home/jaj256/genome/dm3hg38/dm3hg38rDNA"

## Mapq value for filtering multimappers
MAPQ=10

#################################################################
#                           PIPELINE                            #
#################################################################


# Unzipping if needed
echo "unzipping..."
for FILE in fastq/*
    do
        if [[ "$FILE" == *.gz ]]
        then
            gunzip $FILE &
        fi
    done
```

```
wait

# Removing extra info from filenames.
# This is general and works with files from Cornell BRC.
# If filenames are formatted differently, does nothing.
echo "renaming if needed..."
for FILE in $(ls fastq/)
    do
        NEW=fastq/"$(echo "$FILE" |
            sed 's/^[0-9]\+_[0-9]\+_[0-9]\+_[0-9A-Z]\+_//' |
            sed 's/_[ATCG]\{6,8\}_/_/')"
        if [ ! -s "$NEW" ]
        then
            mv fastq/"$FILE" "$NEW"
        fi
    done

mkdir -p logs
mkdir -p logs/fastqc

# Running fastqc on files
echo "running fastqc if needed..."
for FILE in fastq/*.fastq
    do
        if [ ! -s logs/fastqc/"$(basename ${FILE/.fastq/_fastqc.zip})" ]
        then
            fastqc "$FILE" -o logs/fastqc --quiet  &
        fi
    done
wait


mkdir -p trimmedFastq

# Autodetecting paired end files
echo "detecting paired end files..."
NUM=$(ls fastq | wc -l)
NUM_REDUCED=$(ls fastq | sed 's/_R.*//' | uniq | wc -l)
if [[ $NUM == $NUM_REDUCED ]]
    then
        PAIRED="N"
        echo "detected ""$NUM"" single end fastq files. exiting..."
        exit
    else
        PAIRED="Y"
        echo "detected ""$NUM_REDUCED"" paired end fastq files"
fi

# Trimming adapters and filtering rRNA reads
# Four logical branches for 5' and 3' UMI, 5' only, 3' only, and no UMI
echo "trimming adapters and filtering rDNA reads..."
mkdir -p logs/fastp
mkdir -p logs/rRNA
```

```
mkdir -p trimmedFastq
if [[ $PAIRED == "Y" ]]
    then
    # Branches for either 3' UMI or both UMIs
    if [[ $THREEP_UMI == "Y" ]]
        then
        # Branch for both UMIs
        if [[ $FIVEP_UMI == "Y" ]]
            then
            for PAIR in $(ls fastq | sed 's/_R[1-2].*//' | uniq )
                do
                    if [ ! -s trimmedFastq/${PAIR}_R1.fastq ]
                        then
                        echo "trimming adapters and filtering rRNA reads for "${PAIR}
                        (fastp \
                        -i fastq/${PAIR}_R1.fastq \
                        -I fastq/${PAIR}_R2.fastq \
                        --adapter_sequence $ADAPTOR_1 \
                        --adapter_sequence_r2 $ADAPTOR_2 \
                        --umi \
                        --stdout \
                        --umi_loc=per_read \
                        --umi_len=${UMI_LEN} \
                        --html logs/fastp/${PAIR}_fastp.html \
                        -w $(echo ${THREADS}/3 | bc)\
                        -c \
                        --overlap_len_require 15 2> logs/fastp/${PAIR}_fastp.log) |
                        (bowtie2 \
                        --fast-local \
                        --un-conc trimmedFastq/${PAIR}.fastq \
                        --interleaved - \
                        -x ${RDNA} \
                        --threads $(echo ${THREADS}/3*2 | bc)
                        2> logs/rRNA/${PAIR}_rRNA_bowtie.log) > /dev/null
                    fi
                done
        # Branch for just 3' UMI
            else
            for PAIR in $(ls fastq | sed 's/_R[1-2].*//' | uniq )
                do
                    if [ ! -s trimmedFastq/${PAIR}_R1.fastq ]
                    then
                    echo "trimming adapters and filtering rRNA reads for "${PAIR}
                    (fastp \
                    -i fastq/${PAIR}_R1.fastq \
                    -I fastq/${PAIR}_R2.fastq \
                    --adapter_sequence $ADAPTOR_1 \
                    --adapter_sequence_r2 $ADAPTOR_2 \
                    --umi \
                    --stdout \
                    --umi_loc=read1 \
                    --umi_len=${UMI_LEN} \
                    --html logs/fastp/${PAIR}_fastp.html \
```

```
                -w $(echo ${THREADS}/3 | bc) \
                -c \
                --overlap_len_require 15 2> logs/fastp/${PAIR}_fastp.log) |
                (bowtie2 \
                --fast-local \
                --un-conc trimmedFastq/${PAIR}.fastq \
                --interleaved - \
                -x ${RDNA} \
                --threads $(echo ${THREADS}/3*2 | bc)
                2> logs/rRNA/${PAIR}_rRNA_bowtie.log) > /dev/null
                fi
            done
        fi
    # Branch for only 5' UMI or no UMIs
    else
    # Branch for only 5' UMI
    if [[ $FIVEP_UMI == "Y" ]]
        then
        for PAIR in $(ls fastq | sed 's/_R[1-2].*//' | uniq )
            do
                if [ ! -s trimmedFastq/${PAIR}_R1.fastq ]
                    then
                    echo "trimming adapters and filtering rRNA reads for "${PAIR}
                    (fastp \
                    -i fastq/${PAIR}_R1.fastq \
                    -I fastq/${PAIR}_R2.fastq \
                    --adapter_sequence $ADAPTOR_1 \
                    --adapter_sequence_r2 $ADAPTOR_2 \
                    --umi \
                    --stdout \
                    --umi_loc=read2 \
                    --umi_len=${UMI_LEN} \
                    --html logs/fastp/${PAIR}_fastp.html \
                    -w $(echo ${THREADS}/3 | bc) \
                    -c \
                    --overlap_len_require 15 2> logs/fastp/${PAIR}_fastp.log) |
                    (bowtie2 \
                    --fast-local \
                    --un-conc trimmedFastq/${PAIR}.fastq \
                    --interleaved - \
                    -x ${RDNA} \
                    --threads $(echo ${THREADS}/3*2 | bc)
                    2> logs/rRNA/${PAIR}_rRNA_bowtie.log) > /dev/null
                fi
            done
            # Branch for no UMI
            else
                for PAIR in $(ls fastq | sed 's/_R[1-2].*//' | uniq )
                do
                    if [ ! -s trimmedFastq/${PAIR}_R1.fastq ]
                        then
                        echo "trimming adapters and filtering rRNA reads for "${PAIR}
                        (fastp \
```

```
                            -i fastq/${PAIR}_R1.fastq \
                            -I fastq/${PAIR}_R2.fastq \
                            --adapter_sequence $ADAPTOR_1 \
                            --adapter_sequence_r2 $ADAPTOR_2 \
                            --stdout \
                            --html logs/fastp/${PAIR}_fastp.html \
                            -w $(echo ${THREADS}/3 | bc) \
                            -c \
                            --overlap_len_require 15 2> logs/fastp/${PAIR}_fastp.log) |
                            (bowtie2 \
                            --fast-local \
                            --un-conc trimmedFastq/${PAIR}.fastq \
                            --interleaved - \
                            -x ${RDNA} \
                            --threads $(echo ${THREADS}/3*2 | bc)
                            2> logs/rRNA/${PAIR}_rRNA_bowtie.log) > /dev/null
                        fi
                    done
                fi
            fi
        fi

# Cleaning up filenames in trimmedFastq (bowtie automatically names PE --un output)
for FILE in trimmedFastq/*1.fastq
do
    if [ ! -s ${FILE/.1.fastq/_R1.fastq} ]
    then
        mv "$FILE" ${FILE/.1.fastq/_R1.fastq}
    fi
done

for FILE in trimmedFastq/*2.fastq
do
    if [ ! -s ${FILE/.2.fastq/_R2.fastq} ]
    then
        mv "$FILE" ${FILE/.2.fastq/_R2.fastq}
    fi
done

# Aligning to spike in genome to get normalization factors
mkdir -p spikeBAM
mkdir -p logs/spikeAlign

if [[ "$PAIRED" == "Y" ]]
then
    for PAIR in $(ls trimmedFastq | sed 's/_R[1-2].*//' | uniq )
    do
        if [ ! -s "spikeBAM/${PAIR}_hg38.BAM" ]
        then
            echo "aligning ${PAIR} to spike in genome"
            (bowtie2 \
            --local \
            --very-sensitive-local \
```

```
            --threads $(echo ${THREADS}/3*2 | bc) \
            --no-unal \
            --no-mixed \
            --no-discordant \
            -x "$GENOME_SPIKE" \
            -1 "trimmedFastq/${PAIR}_R1.fastq" \
            -2 "trimmedFastq/${PAIR}_R2.fastq" \
            2> logs/spikeAlign/${PAIR}_spikeAlign.log) |
            samtools view -hS -f 2 -q ${MAPQ} |
            perl -n -e 'print $_ if (/^\@/ || /'${SPIKE_PREFIX}'/ ) ' |
            samtools view -b |
            samtools sort -@ $(echo ${THREADS}/3 | bc) -o spikeBAM/${PAIR}.BAM
            samtools index spikeBAM/${PAIR}.BAM
        fi
    done
fi

# Aligning to experimental genome
mkdir -p BAM
mkdir -p logs/align

if [[ "$PAIRED" == "Y" ]]
then
    for PAIR in $(ls trimmedFastq | sed 's/_R[1-2].*//' | uniq )
    do
        if [ ! -s "BAM/${PAIR}.BAM" ]
        then
            echo "aligning ${PAIR} to experimental genome"
            (bowtie2 \
            --local \
            --sensitive-local \
            --threads $(echo ${THREADS}/3*2 | bc) \
            -x "$GENOME_EXP" \
            -1 "trimmedFastq/${PAIR}_R1.fastq" \
            -2 "trimmedFastq/${PAIR}_R2.fastq" \
            2> logs/align/${PAIR}_align.log) |
            samtools view -bS -f 2 -q ${MAPQ} |
            samtools sort -@ $(echo ${THREADS}/3 | bc) -o BAM/${PAIR}.BAM
            samtools index BAM/${PAIR}.BAM
        fi
    done
fi

# Deduplicating with UMIs (experimental BAM)
mkdir -p BAMdeDuped
mkdir -p logs/deDup

for FILE in BAM/*.BAM
do
    if [ ! -s "BAMdeDuped/$(basename ${FILE%.BAM}_deDuped.BAM)" ]
    then
        (umi_tools dedup \
        -I "$FILE" \
```

```
        --umi-separator=":" \
        --paired \
        -S "BAMdeDuped/$(basename ${FILE%.BAM}_deDuped.BAM)" \
        )> "logs/deDup/$(basename ${FILE%.BAM}_deDup.log)" &&
        samtools index "BAMdeDuped/$(basename ${FILE%.BAM}_deDuped.BAM)"
    fi
done


# Deduplicating with UMIs (Spike-In BAM)
mkdir -p spikeBAMdeDuped
mkdir -p logs/spikedeDup

for FILE in spikeBAM/*.BAM
do
    if [ ! -s "spikeBAMdeDuped/$(basename ${FILE%.BAM}_deDuped.BAM)" ]
    then
        (
        umi_tools dedup \
        -I "$FILE" \
        --paired \
        --umi-separator=":" \
        -S "spikeBAMdeDuped/$(basename ${FILE%.BAM}_deDuped.BAM)" \
        )> "logs/spikedeDup/$(basename ${FILE%.BAM}_deDup.log)" &&
        samtools index "spikeBAMdeDuped/$(basename ${FILE%.BAM}_deDuped.BAM)"
    fi
done

# Generating table of Alignment metrics
mkdir -p info

if [ ! -s info/infoTable.tsv ]
then
touch info/infoTable.tsv
echo -e Name'\t'\
        RawReads'\t'\
        NonDimerReads'\t'\
        %dimer'\t'\
        insertSize'\t'\
        rRNAreads'\t'\
        %rRNA'\t'\
        passedFilters'\t'\
        bowtieConcordant'\t'\
        bowtieMulti'\t'\
        bowtieUnal'\t'\
        bowtieOverallMap%'\t'\
        bowtieConcordant%'\t'\
        bowtieMulti%'\t'\
        bowtieUnal%'\t'\
        uniqueMapped'\t'\
        uniqueMappedNondup'\t'\
        %PCRdups'\t'\
        uniqueMappedSpikein'\t'\
```

```
        uniqueMappedSpikeinNondup'\t'\
        spikeInPCRdups% >> info/infoTable.tsv

for SAMPLE in $(ls BAM/*.BAM | sed 's/.BAM//' | sed 's/BAM\///' )
do
    NAME=${SAMPLE}
    RAW_READS=$(cat logs/fastp/${SAMPLE}_fastp.log |
                grep "total reads:" | head -n 1 |
                awk '{print $3}')
    TRIMMED_READS=$(cat logs/fastp/${SAMPLE}_fastp.log |
                grep "total reads:" | tail -n 1 |
                awk '{print $3}')
    PER_DIMER=$(echo "(1-"${TRIMMED_READS}"/"${RAW_READS}")*100" | bc -l)%
    INSERT_SIZE=$(cat logs/fastp/${SAMPLE}_fastp.log |
                grep "Insert size peak" |
                awk '{print $8}')
    PASSED_FILTERS=$(cat logs/align/${SAMPLE}_align.log |
                grep "reads; of these:$" |
                awk '{print $1}')
    RRNA=$(echo ${TRIMMED_READS}"-"${PASSED_FILTERS} | bc )
    PER_RRNA=$(echo ${RRNA}"/"${RAW_READS}"*100" | bc -l)%
    B_CONC=$(cat logs/align/${SAMPLE}_align.log |
            grep "aligned concordantly exactly 1 time$" |
            awk '{print $1}')
    B_MULTI=$(cat logs/align/${SAMPLE}_align.log |
            grep "aligned concordantly >1 times$" |
            awk '{print $1}')
    B_UNAL=$(cat logs/align/${SAMPLE}_align.log |
            grep "aligned concordantly 0 times$" |
            awk '{print $1}')
    B_OAP=$(cat logs/align/${SAMPLE}_align.log |
            grep "overall alignment rate$" |
            awk '{print $1}')
    B_CONC_PER=$(echo ${B_CONC}"/"${PASSED_FILTERS}"*100" | bc -l)%
    B_MULTI_PER=$(echo ${B_MULTI}"/"${PASSED_FILTERS}"*100" | bc -l)%
    B_UNAL_PER=$(echo ${B_UNAL}"/"${PASSED_FILTERS}"*100" | bc -l)%
    UNIQ_MAPPED=$(cat logs/deDup/${SAMPLE}_deDup.log |
            grep "Input Reads:" | awk '{print $10}')
    UNIQ_MAPPED_DEDUP=$(cat logs/deDup/${SAMPLE}_deDup.log |
            grep "Number of reads out:" | awk '{print $8}')
    PER_DUPS=$(echo "(1-"${UNIQ_MAPPED_DEDUP}"/"${UNIQ_MAPPED}")*100" | bc -l)%
    UNIQ_MAPPED_SPIKE=$(cat logs/spikedeDup/${SAMPLE}_deDup.log |
            grep "Input Reads:" | awk '{print $10}')
    UNIQ_MAPPED_DEDUP_SPIKE=$(cat logs/spikedeDup/${SAMPLE}_deDup.log |
            grep "Number of reads out:" | awk '{print $8}')
    PER_DUPS_SPIKE=$(echo "(1-"${UNIQ_MAPPED_DEDUP_SPIKE}"/"${UNIQ_MAPPED_SPIKE}")*100" |
            bc -l)%

    echo -e $NAME'\t'\
    $RAW_READS'\t'\
    $TRIMMED_READS'\t'\
    $PER_DIMER'\t'\
    $INSERT_SIZE'\t'\
```

```
    $RRNA'\t'\
    $PER_RRNA'\t'\
    $PASSED_FILTERS'\t'\
    $B_CONC'\t'\
    $B_MULTI'\t'\
    $B_UNAL'\t'\
    $B_OAP'\t'\
    $B_CONC_PER'\t'\
    $B_MULTI_PER'\t'\
    $B_UNAL_PER'\t'\
    $UNIQ_MAPPED'\t'\
    $UNIQ_MAPPED_DEDUP'\t'\
    $PER_DUPS'\t'\
    $UNIQ_MAPPED_SPIKE'\t'\
    $UNIQ_MAPPED_DEDUP_SPIKE'\t'\
    $PER_DUPS_SPIKE  >> info/infoTable.tsv

done
fi

# Making non-normalized bigWig files
mkdir -p bw
for FILE in BAMdeDuped/*.BAM
do
    if [ ! -s "bw/$(basename ${FILE/.BAM/_fwd.bw})" ]
    then
        bamCoverage \
        --bam $FILE \
        --skipNonCoveredRegions \
        --outFileName bw/$(basename ${FILE/.BAM/_fwd.bw}) \
        --binSize 1 \
        --numberOfProcessors ${THREADS} \
        --normalizeUsing None \
        --Offset 1 \
        --samFlagInclude 82
    fi
    if [ ! -s "bw/$(basename ${FILE/.BAM/_rev.bw})" ]
    then
        bamCoverage \
        --bam $FILE \
        --skipNonCoveredRegions \
        --outFileName bw/$(basename ${FILE/.BAM/_rev.bw}) \
        --binSize 1 \
        --numberOfProcessors ${THREADS} \
        --normalizeUsing None \
        --Offset 1 \
        --samFlagInclude 98
    fi
done
```

Supplementary Code 2. ATAC-seq alignment pipeline

```
#!/bin/bash
# This script is for aligning/peak calling/making bigwig signal tracks for ATAC-seq data

# Number of threads to use for applications that support multithreading
THREADS=50

# Unzipping if needed
for FILE in fastq/*
    do gunzip $FILE -q &
done

# Renaming files
for FILE in $(ls fastq/)
    do
        NEW=fastq/"$(echo "$FILE" |
        sed 's/^[0-9]\+_[0-9]\+_[0-9]\+_[0-9A-Z]\+_//' |
        sed 's/_[ATCG]\{6,8\}_/_/')"
        if [ ! -s "$NEW" ]
        then
        mv fastq/"$FILE" "$NEW"
        fi
    done

# Running Fastqc
mkdir -p logs
mkdir -p logs/fastqc

for FILE in fastq/*.fastq
do
    if [ ! -s logs/fastqc/"$(basename ${FILE/.fastq/_fastqc.zip})" ]
    then
        fastqc "$FILE" -o logs/fastqc/ --quiet  &
    fi
done
wait

# Aligning files
mkdir -p BAM
mkdir -p logs/align

for PAIR in $(ls fastq | sed 's/_R[1-2].*//' | uniq)
do
    if [ ! -s "BAM/${PAIR}.BAM" ]
    then
        (bowtie2 --local --very-sensitive-local --threads ${THREADS} \
        --no-unal -I 10 -X 1000 -x ~/genome/dm6/dm6Hsp70AaOnly \
        -1 fastq/${PAIR}_R1.fastq \
        -2 fastq/${PAIR}_R2.fastq \
        2> logs/align/${PAIR}_align.log) |
        samtools view -bS -q 10 -f 2 |
        samtools sort -o BAM/${PAIR}.BAM
```

```
        samtools index BAM/${PAIR}.BAM
    fi
done

# Plotting coverage of reads < 120 bp insert (ATAC Hypersensitivity)
mkdir -p bwDHS

for FILE in BAM/*.BAM
do
    bamCoverage \
    --bam ${FILE} \
    --outFileName bwDHS/$(basename ${FILE/.BAM/_AtacDHS.bw}) \
    --binSize 1 \
    --numberOfProcessors ${THREADS} \
    --normalizeUsing None \
    --skipNAs \
    --extendReads \
    --maxFragmentLength 120
done


# Plotting coverage of centers of reads  200 > insert > 130 (mononucleosomes)
mkdir -p bwMonoNucs
for FILE in repMergedBAM/*.BAM
do
    if [ ! -s bwMonoNucs/$(basename ${FILE/.BAM/_AtacMonoNucs.bw}) ]
    then
        bamCoverage \
        --bam ${FILE} \
        --outFileName bwMonoNucs/$(basename ${FILE/.BAM/_AtacMonoNucs.bw}) \
        --binSize 1 \
        --MNase \
        --numberOfProcessors ${THREADS} \
        --normalizeUsing None
    fi
done

# Calling peaks on all BAM files (pooling all samples)
source /programs/bin/util/setup_macs2.sh

mkdir -p peaks
macs2 callpeak \
  -t BAM/*.BAM \
  -f BAMPE \
  -g dm \
  -n "ALL_ATAC_PEAKS" \
  --outdir peaks \
  --call-summits
```

Supplementary Code 3. 3' RNA-seq alignment pipeline

```
#!/bin/bash

# Number of threads to use for applications that support multithreading
THREADS=50

# building STAR index
/programs/STAR/STAR \
    --runThreadN ${THREADS} \
    --runMode genomeGenerate \
    --genomeDir /workdir/jaj256/gaf/new/RNAseq/STARindex \
    --genomeFastaFiles ~/genome/dm6hg38ERCC/dm6hg38ERCC.fa \
    --sjdbGTFfile ~/genome/dm6hg38ERCC/dm6hg38ERCC.gtf \
    --sjdbOverhang 68


mkdir -p trimmedFastq
mkdir -p logs
mkdir -p logs/fastp

# Trimming Adapters and polyA sequences, and extracting UMIs
for FILE in fastq/*.fastq
    do
        fastp \
            -i ${FILE} \
            -o trimmedFastq/$(basename ${FILE}) \
            --length_required 25 \
            --adapter_fasta adapters.fa \
            --umi \
            --umi_loc=read1 \
            --umi_len=6 \
            --html logs/fastp/$(basename ${FILE/.fastq/_fastp.html}) \
            -w ${THREADS} 2> logs/$(basename ${FILE/.fastq/_fastp.log})
    done

# Aligning to the combined experimental/Spike-in genome
mkdir -p BAM
mkdir -p logs/STAR
for FILE in trimmedFastq/*.fastq
    do
        /programs/STAR/STAR \
            --runThreadN ${THREADS} \
            --genomeDir STARindex/ \
            --readFilesIn ${FILE} \
            --outFilterType BySJout \
            --outFilterMultimapNmax 20 \
            --alignSJoverhangMin 8 \
            --alignSJDBoverhangMin 1 \
            --outFilterMismatchNmax 999 \
            --outFilterMismatchNoverLmax 0.1 \
            --alignIntronMin 20 \
            --alignIntronMax 1000000 \
```

```
            --alignMatesGapMax 1000000 \
            --outSAMattributes NH HI NM MD \
            --outSAMtype BAM SortedByCoordinate \
            --outFileNamePrefix BAM/$(basename ${FILE/.fastq/}) \
            2> logs/STAR/$(basename ${FILE/.fastq/_STAR.log})
        done

# Deduplicating using UMIs
mkdir -p BAMdeDuped
mkdir -p logs/deDup
for FILE in BAM/*.BAM
do
    umi_tools dedup \
    -I ${FILE} \
    -S "BAMdeDuped/$(basename ${FILE})" \
    --umi-separator=":" \
    > logs/deDup/$(basename ${FILE%.BAM}_deDup.log) &
done
wait

for FILE in BAMdeDuped/*.BAM
    do samtools index $FILE &
    done
    wait

# Splitting spike-in and experimental alignments
mkdir -p dm6BAM
mkdir -p ERCCBAM

for FILE in BAMdeDuped/*.BAM
    do
    samtools view -hS -q 255 -F 4 ${FILE} |
    perl -n -e 'print $_ if (/^\@/ || /ERCC/) ' |
    samtools view -b |
    samtools sort -o ERCCBAM/$(basename ${FILE})

    samtools view -hS -q 255 -F 4 ${FILE} |
    perl -n -e 'print $_ if (/^\@/ || !(/hg38/ || /ERCC/)) ' |
    samtools view -b |
    samtools sort -o dm6BAM/$(basename ${FILE})
    done

# Counting # of alignments to experimental and spike-in genomes
mkdir -p spikeCounts
touch spikeCounts/spikeCounts.tsv
echo -e sample"\t"dm6reads"\t"ERCCreads > spikeCounts/spikeCounts.tsv

for FILE in BAMdeDuped/*.BAM;
do
    FILENAME=$(basename ${FILE%.BAM})
    DM6=$(samtools view -c dm6BAM/${FILENAME}.BAM)
    ERCC=$(samtools view -c ERCCBAM/${FILENAME}.BAM)
    echo -e ${FILENAME}"\t"${DM6}"\t"${ERCC} >> spikeCounts/spikeCounts.tsv
```

```
done

# Making bw signal tracks (non-normalized)
for FILE in dm6BAM/*.BAM
do
    bamCoverage \
    --bam $FILE \
    --skipNonCoveredRegions \
    --outFileFormat bigwig \
    --outFileName bdg/$(basename ${FILE/.BAM/_fwd.bw}) \
    --binSize 1 \
    --numberOfProcessors ${THREADS} \
    --normalizeUsing None \
    --Offset 1 \
    --filterRNAstrand reverse
    bamCoverage \
    --bam $FILE \
    --skipNonCoveredRegions \
    --outFileFormat bigwig \
    --outFileName bdg/$(basename ${FILE/.BAM/_rev.bw}) \
    --binSize 1 \
    --numberOfProcessors ${THREADS} \
    --normalizeUsing None \
    --Offset 1 \
    --filterRNAstrand forward
done
```

Supplementary Code 4. CUT&RUN alignment pipeline

```
#!/bin/bash

# Number of threads to use for applications that support multithreading
THREADS=50

# Running Fastqc
mkdir -p logs
mkdir -p logs/fastqc

    for FILE in fastq/*.fastq
        do
            if [ ! -s logs/fastqc/$(basename ${FILE/.fastq/_fastqc.zip}) ]
                then
                fastqc ${FILE} -q -o logs/fastqc &
            fi
        done
        wait

# Trimming adapters, aligning, sorting, and indexing
mkdir -p BAM
mkdir -p logs/align
mkdir -p logs/fastp

for PAIR in $(ls fastq | sed 's/_R[1-2].*//' | uniq)
    do
        if [ ! -s BAM/${PAIR}.BAM ]
            then
            (fastp \
            -i fastq/${PAIR}_R1.fastq \
            -I fastq/${PAIR}_R2.fastq \
            --adapter_sequence AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC \
            --adapter_sequence_r2 \
            AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTAGATCTCGGTGGTCGCCGTATCATT \
            --stdout \
            --html logs/fastq/${PAIR}_fastp.log \
            -w $(echo ${THREADS}/3 | bc) \
            -c --overlap_len_require 15
            2> logs/fastp/${PAIR}_fastp.log) |
            (bowtie2 \
            --very-sensitive-local \
            --no-unal --no-discordant \
            --interleaved - \
            -x ~/genome/dm6/dm6Hsp70AaOnly \
            --threads $(echo ${THREADS}/3*2 | bc)
            2> logs/align/${PAIR}_align.log) |
            samtools view -bS -f 2 -q 10 |
            samtools sort -o BAM/${PAIR}.BAM
            samtools index BAM/${PAIR}.BAM
        fi
    done
```

```
# Making bigWig files
mkdir -p bw
for FILE in BAM/*.BAM;
do
  if [ ! -s bw/$(basename ${FILE/BAM/bw}) ]
  then
    bamCoverage \
      -b $FILE \
      -o bw/$(basename ${FILE/BAM/bw}) \
      --binSize 10 \
      -p ${THREADS} \
      --normalizeUsing None \
      --skipNonCoveredRegions \
      --extendReads \
      --maxFragmentLength 120
fi
done
```

Supplementary Code 5. ChIP-seq realignment pipeline

```
#!/bin/bash

# Number of threads to use for applications that support multithreading
THREADS=50

# Aligning
mkdir -p logs/align
mkdir -p BAM
for FILE in fastq/*.fastq
do
  (bowtie2 \
    --very-sensitive-local \
    --no-unal \
    -x ~/genome/dm6/dm6Hsp70AaOnly \
    --threads ${THREADS} \
    -U ${FILE} 2> logs/align/$(basename ${FILE%.fastq})_align.log) |
    samtools view -bS -q 10 |
    samtools sort -o BAM/$(basename ${FILE%.fastq}).BAM
    samtools index BAM/$(basename ${FILE%.fastq}).BAM
done

# Making bigWig files
mkdir -p bw
for FILE in BAM/*.BAM
do
  bamCoverage \
    -b ${FILE} \
    -o bw/$(basename ${FILE/.BAM/.bw}) \
    --binSize 10 \
    --extendReads 200 \
    --normalizeUsing None \
    --skipNonCoveredRegions
done
```

Supplementary Code 6. Data Analysis (R Scripts)

All code chunks in this section were run in RStudio.
See Supplementary Code 7 for version of all packages used.

Loading packages

```
library(tidyverse)
library(DESeq2)
library(ggpubr)
library(viridis)
library(scales)
library(rtracklayer)
library(GenomicRanges)
library(BiocParallel)
library(BRGenomics)
library(extrafont)
library(patchwork)
library(plyr)
library(ComplexHeatmap)
library(circlize)
library(tiff)
library(eulerr)
library(gridExtra)
loadfonts()
source("/Users/julius/Google Drive/R/customPackages/browserPlotR.R")
```

Functions

Custom R functions used throughout the remainder of the code chunks

```
fc_corr.jj <-
  function(res1,
           res2,
           pval = 0.01,
           cols = c("#BB0021", "#3B4992", "#BBBBBB")) {
    # This function takes the path to two saved DESeq2 Results files (.tsv)
    # Parses them, separates them into activated and repressed classes (padj < 0.05)
    # and plots a scatter of res1 l2FC by res2 l2FC
    # with a glm fit (shaded CI = 95%) of each class
    # significance calling is based on the pvalue of res1

    res.df <- data.frame(
      res1_l2FC = as.data.frame(res1)$log2FoldChange,
      res1_padj = as.data.frame(res1)$padj,
      res2_l2FC = as.data.frame(res2)$log2FoldChange,
      res2_padj = as.data.frame(res2)$padj
    )

    res.df <- res.df[!is.na(res.df$res1_padj) & !is.na(res.df$res2_l2FC),]
```

```
res.df$sig <- ifelse((res.df$res1_padj < pval),
                     ifelse(res.df$res1_l2FC < 0,
                            "down",
                            "up"),
                     "ns"
)

res.df$sig <- factor(res.df$sig, levels = c("up", "down", "ns"))

num_up <- nrow(filter(res.df, sig == "up"))
num_down <- nrow(filter(res.df, sig == "down"))
max_y <- max(res.df$res2_l2FC)
min_y <- min(res.df$res2_l2FC)
max_x <- max(res.df$res1_l2FC)
min_x <- min(res.df$res1_l2FC)


p <-
    ggplot(res.df,
           aes(x = res1_l2FC, y = res2_l2FC)) +
    geom_abline(intercept = 0, slope = 1) +
    geom_point(
        data = filter(res.df, sig == "ns"),
        size = 0.75,
        alpha = 0.75,
        stroke = 0,
        show.legend = F,
        color = cols[3]
    ) +
    geom_point(
        data = filter(res.df, sig == "up"),
        size = 0.75,
        alpha = 0.5,
        stroke = 0,
        show.legend = F,
        color = cols[1]
    ) +
    geom_point(
        data = filter(res.df, sig == "down"),
        size = 1,
        alpha = 0.75,
        stroke = 0,
        show.legend = F,
        color = cols[2]
    ) +
    geom_smooth(
        data = filter(res.df, sig == "down"),
        method = "glm",
        level = 0.95,
        size = 0.5,
        color = "black",
        show.legend = F
    ) +
```

```
        geom_smooth(
            data = filter(res.df, sig == "up"),
            method = "glm",
            level = 0.95,
            size = 0.5,
            color = "black",
            show.legend = F
        ) +
        xlab(paste0(deparse(substitute(res1)), "log2 FC")) +
        ylab(paste0(deparse(substitute(res2)), "log2 FC")) +
        ggtheme.jj() +
        geom_vline(xintercept = 0, linetype = "dashed", alpha = 0.5)+
        geom_hline(yintercept = 0, linetype = "dashed", alpha = 0.5)+
        annotate(geom = 'text',
                 label = num_up,
                 x = max_x * 0.9,
                 y = min_y * 0.9,
                 color = cols[1]
                )+
        annotate(geom = 'text',
                 label = num_down,
                 x = min_x * 0.9,
                 y = max_y * 0.9,
                 color = cols[2]
                 )

    return(p)
}

subset_DESeq.jj <- function(df,
                            col_data,
                            scale_facts,
                            pos_inc,
                            contrast,
                            formula) {
  # Function that takes a dataframe and calls DESeq2
  # On a subset of the dataframe. Provide a df of counts,
  # a colData object (see DESeq2 Vignette), a vector of
  # scale factors, positions to include as a numeric
  # vector (pos.inc), and a character vector to contrast
  # Returns a DESeq2 results object


  df <- df[, pos_inc]
  col_data <- col_data[pos_inc, , drop = FALSE]
  dds <- DESeqDataSetFromMatrix(countData = df,
                                colData = col_data,
                                design = formula)
  sizeFactors(dds) <- scale_facts[pos_inc]
  dds <- DESeq(dds)
  res <- results(dds,
                 contrast = contrast)
```

```
}

ggMetaplot <- function(meta.mat) {
    # Takes a subsampled matrix (output by metaSubsample in BRGenomics)
    # and produces a 'default' metaplot using ggplot. This can easily be customized
    # using standard ggplot syntax:
    # ggMetaplot(meta.mat) + scale_color_manual(values = c("blue", "red")), for example
    return(
        ggplot(
            meta.mat,
            aes(
                x = x,
                y = mean,
                ymax = upper,
                ymin = lower,
                color = sample.name
            )
        ) +
            geom_line(size = 1, alpha = 1) +
            geom_ribbon(
                alpha = 0.2,
                aes(fill = sample.name),
                color = NA,
                show.legend = T
            ) +
            ggtheme.jj() +
            xlab("Distance from TSS (bp)") +
            ylab("Mean + 75% CI")

    )
}

RPMnorm <- function(gr){
    gr$score <- (1e6 / sum(gr$score)) * gr$score
    return(gr)
}

ggtheme.jj <- function() {
    # Custom theme options for ggplot2 graphics
    theme_classic(base_size=10, base_family="Helvetica") %+replace%
        theme(
            axis.text = element_text(size = 8),
            axis.ticks = element_line(colour = "black"),
            legend.key = element_blank(),
            panel.background = element_rect(fill = "white", colour = NA),
            panel.border = element_blank(),
            panel.grid.major = element_blank(),
            panel.grid.minor = element_blank(),
            strip.background = element_blank(),
            strip.text = element_text(size=10),
            plot.title = element_text(hjust = 0.5, size = 10),
            axis.title = element_text(size = 8, face = "bold")
        )
```

```
}

rep_corr_scatter.jj <-
    function(df,
             sep_colnames_into = c(),
             colsep = "_",
             drop_cols = NA
    ) {
    # wrapper function that takes a dataframe of counts, splits them
    # by replicate, and plots density of points in hexbins, rep1 vs rep2
    # df: dataframe of counts. Must contain at least one column of unique IDs.
    #       each column should have counts for one sample
    # sep_colnames_into: character vector of columns to separate colnames into.
    #       one value MUST be "condition", the other must be "replicate".
    #       List NAs for unused spots, according to colsep. For example,
    #       a column named "BAP170_NHS_Rep1" would be
    #       c("condition", NA, "rep")
    # colsep: field separator for above split
    # drop_cols: vector of column numbers or names to drop, useful if starting df
    #       has other info like start, stop, etc
    # Other attributes that ggplots can have can be added by calling this function
    # then using + ylab() + xlab() etc

    if(!is.na(drop_cols)) {
        df <- df[,-drop_cols]
    }

    df.ltb <- df %>%
        gather("colname", "count", -tx_name) %>%
        separate(colname, into = sep_colnames_into, sep = colsep) %>%
        spread(rep, count)

    p <-ggplot(df.ltb, aes(
            x = Rep1,
            y = Rep2
        )) +
        geom_abline(
            intercept = 0,
            slope = 1,
            linetype = "solid",
            size = 0.5,
            alpha = 0.75
        ) +
        geom_hex(bins = c(25, 25)) +
        stat_cor(
            method = "spearman",
            label.x.npc = c("left"),
            label.y.npc = c("top"),
            output.type = "text",
            hjust = -0.1
        ) +
        scale_x_log10() +
        scale_y_log10() +
```

```
        scale_fill_viridis_c(name = "Density") +
        ggtheme.jj()

    return(p)
    }

maplot.jj <- function(res, padj_cutoff = 0.01, l2fc_cutoff = 0) {
    # This function takes a DESeq2 Results object and plots a pretty MA plot.
    # res = the results object
    # padj_cutoff = significance cutoff (dbl)
    # l2fc_cutoff = log2 fold change cutoff (dbl)

    resdf <- as.data.frame(res)
    resdf$name <- row.names(resdf)
    resdf[is.na(resdf$padj),]$padj <- 1
    resdf <-
        drop_na(mutate(resdf, "class" = if_else(
            padj < padj_cutoff,
            if_else(log2FoldChange > 0, "Activated", "Repressed"),
            "Unchanged"
        )))

    resdf$class[which(abs(resdf$log2FoldChange) < l2fc_cutoff)] <- "Unchanged"

    numAct <- nrow(filter(resdf, class == "Activated"))
    numRep <- nrow(filter(resdf, class == "Repressed"))
    numUch <- nrow(filter(resdf, class == "Unchanged"))
    max_y <- max(resdf$log2FoldChange)
    min_y <- min(resdf$log2FoldChange)
    max_x <- max(resdf$baseMean)
    min_x <- min(resdf$baseMean)
    x_lim_left <- 10^floor(log10(min_x))
    x_lim_right <- 10^ceiling(log10(max_x))
    y_lim_top <- round_any(max_y, 2, ceiling)
    y_lim_bottom <- round_any(min_y, 2, floor)

    lseq <- function(from, to, length.out) {
        # logarithmic spaced sequence
        # blatantly stolen from library("emdbook"), because need only this
        exp(seq(log(from), log(to), length.out = length.out))
    }

    breaks_x <- lseq(
        from = x_lim_left,
        to = x_lim_right,
        length.out = (log10(x_lim_right) - log10(x_lim_left) + 1)
    )

    breaks_y <- seq(y_lim_bottom, y_lim_top, 2)

    p1 <-
        ggplot(resdf, aes(x = baseMean, y = log2FoldChange, color = class)) +
        geom_point(stroke = 0,
```

```
                alpha = 0.75,
                size = 0.75,
                show.legend = F) +
    xlab('log10 Mean Expression') +
    ylab('log2 Fold Change') +
    scale_x_log10(
        limits = c(x_lim_left, x_lim_right),
        breaks = breaks_x,
        labels = round(log10(breaks_x)),
        expand = c(0,0)
            ) +
    scale_y_continuous(
        limits = c(y_lim_bottom, y_lim_top),
        expand = c(0,0),
        breaks = breaks_y,
        labels = breaks_y
        )+
    scale_color_manual(
        values = c("Activated" = "#BB0021",
                   "Repressed" = "#3B4992",
                   "Unchanged" = "gray"),
        )+
    geom_hline(yintercept = 0,
                size = 0.5,
                alpha = 1) +
    annotate(geom = 'text',
             label = numAct,
             x = min_x,
             y = max_y,
             color = "#BB0021",
             hjust = -0.25,
             vjust = 0.5)+
    annotate(geom = 'text',
             label = numRep,
             x = min_x,
             y = min_y,
             color = "#3B4992",
             hjust = -0.25,
             vjust = -0.5)+
    ggtheme.jj() +
    theme()

if(l2fc_cutoff > 0){
    p1 <- p1+
        geom_hline(yintercept = l2fc_cutoff,
                size = 0.5,
                color = "grey",
                linetype = "dashed")+
  geom_hline(yintercept = -l2fc_cutoff,
             size = 0.5,
             color = "grey",
             linetype = "dashed")
}
```

```
    return(p1)
}
```

Global objects

Scale factors, palettes, color scales, gene list, ATAC/ChIP peaks

```
# Header for reading in DESeq2 results files using read_tsv()
DESeq2_headers.chr <-
    c("name",
      "baseMean",
      "log2FoldChange",
      "lfcSE",
      "stat",
      "pvalue",
      "padj")

# Global palettes and ggplot color scales
RNAi_cols.chr <- c("#BBBBBB", "#66CCEE", "#228833", "#CCBB44", "#AA3377")
updown_cols.chr <- c("#BB0021", "#3B4992")

gg_color_scale_RNAi_ATACDHS <- scale_color_manual(
        values = RNAi_cols.chr,
        name = "RNAi",
        labels = c(
            "LACZ_ATACDHS" = "LACZ",
            "GAF_ATACDHS" = "GAF",
            "BAP170_ATACDHS" = "BAP",
            "NURF301_ATACDHS" = "NURF",
            "NURF301BAP170_ATACDHS" = "NURF+BAP"
        ))

gg_fill_scale_RNAi_ATACDHS <- scale_fill_manual(
        values = RNAi_cols.chr,
        name = "RNAi",
        labels = c(
            "LACZ_ATACDHS" = "LACZ",
            "GAF_ATACDHS" = "GAF",
            "BAP170_ATACDHS" = "BAP",
            "NURF301_ATACDHS" = "NURF",
            "NURF301BAP170_ATACDHS" = "NURF+BAP"
        ))

gg_color_scale_RNAi_ATACMN <- scale_color_manual(
        values = RNAi_cols.chr,
        name = "RNAi",
        labels = c(
            "LACZ_ATACMN" = "LACZ",
            "GAF_ATACMN" = "GAF",
            "BAP170_ATACMN" = "BAP",
            "NURF301_ATACMN" = "NURF",
            "NURF301BAP170_ATACMN" = "NURF+BAP"
```

```
        ))

gg_fill_scale_RNAi_ATACMN <- scale_fill_manual(
        values = RNAi_cols.chr,
        name = "RNAi",
        labels = c(
            "LACZ_ATACMN" = "LACZ",
            "GAF_ATACMN" = "GAF",
            "BAP170_ATACMN" = "BAP",
            "NURF301_ATACMN" = "NURF",
            "NURF301BAP170_ATACMN" = "NURF+BAP"
        ))

gg_color_scale_RNAi_PROseq <- scale_color_manual(
        values = RNAi_cols.chr,
        name = "RNAi",
        labels = c(
            "LACZ_PROseq" = "LACZ",
            "GAF_PROseq" = "GAF",
            "BAP170_PROseq" = "BAP",
            "NURF301_PROseq" = "NURF",
            "NURF301BAP170_PROseq" = "NURF+BAP"
        ))

gg_fill_scale_RNAi_PROseq <- scale_fill_manual(
        values = RNAi_cols.chr,
        name = "RNAi",
        labels = c(
            "LACZ_PROseq" = "LACZ",
            "GAF_PROseq" = "GAF",
            "BAP170_PROseq" = "BAP",
            "NURF301_PROseq" = "NURF",
            "NURF301BAP170_PROseq" = "NURF+BAP"
        ))

# PRO-cap corrected gene list
genes.tbl <- read_tsv(
    "genome/dm6_genes_PROcap_corrected.bed",
    col_names = c(
        "chr",
        "start",
        "stop",
        "gene_name",
        "tx_name",
        "strand",
        "pro_cap_shift",
        "pro_cap_signal"
    )
) %>%
    filter(stop - start > 500) # Filter transcripts less than 500 bp long

# PRO-cap TSS correction occasionally condenses two isoforms that have
# closeby TSSs into one.
```

```
# Remove these becasue they now have duplicate coordinates
# (Brings list from 14818 transcripts to 14474 genes)
genes.tbl <- genes.tbl[which(!duplicated(genes.tbl[, 1:4])), ]

# Making GRanges of genes
genes.gr <-
  tidyChromosomes(
    makeGRangesFromDataFrame(genes.tbl[,-c(7,8)],
                             keep.extra.columns = T))

# ATAC-seq peak summits (called with macs2 using all samples as input)
ATACseq_summits.tbl <- read_tsv(
  "bed/ATACseq_peaks_summits_macs2.bed",
  col_names = c("chr",
                "start",
                "stop",
                "peak_name",
                "score")
)

ATACseq_summits.gr <-
  tidyChromosomes(
    makeGRangesFromDataFrame(ATACseq_summits.tbl,
                             keep.extra.columns = T))

# GAF ChIP-seq peaks
GAF_peaks.tbl <- read_tsv(
  "bed/GAF_ChIPseq_peaks.narrowPeak",
  col_names = c(
    "chr", "start", "end", "name",
    "displayScore", "strand", "summitFC",
    "mlog10p", "mlog10q", "summitPosition"
  )
)

GAF_peaks.gr <- tidyChromosomes(
  makeGRangesFromDataFrame(GAF_peaks.tbl, keep.extra.columns = T)
)

# PRO-seq scaling factors (derived from human spike-in cells)
# All scale factors are in the format of 1 / x.xxx because DESeq2 divides
# the matrix by the scale factor. These scale factors were designed calculated
# based on scaling by multiplication, so the inverse is provided here for
# consistancy and direct compatibility with DESeq2
samples_PROseq.chr <-     c(
    "BAP170_PROseq_Rep1",
    "BAP170_PROseq_Rep2",
    "GAF_PROseq_Rep1",
    "GAF_PROseq_Rep2",
    "LACZ_PROseq_Rep1",
    "LACZ_PROseq_Rep2",
    "NURF301BAP170_PROseq_Rep1",
    "NURF301BAP170_PROseq_Rep2",
```

```
    "NURF301_PROseq_Rep1",
    "NURF301_PROseq_Rep2"
)
scale_facts_PROseq.dbl <-     c(
    1 / 0.626244788,
    1 / 1.000000000,
    1 / 0.364059812,
    1 / 0.650773152,
    1 / 0.533494922,
    1 / 0.579464988,
    1 / 0.522119671,
    1 / 0.556802652,
    1 / 0.559650456,
    1 / 0.964053768
)
scale_facts_PROseq.lst <-
  setNames(as.list(scale_facts_PROseq.dbl), samples_PROseq.chr)

# RNA-seq scaling factors (ERCC spike-in derived)
samples_RNAseq.chr <-     c(
    "BAP170_RNAseq_Rep1",
    "BAP170_RNAseq_Rep2",
    "GAF_RNAseq_Rep1",
    "GAF_RNAseq_Rep2",
    "LACZ_RNAseq_Rep1",
    "LACZ_RNAseq_Rep2",
    "NURF301BAP170_RNAseq_Rep1",
    "NURF301BAP170_RNAseq_Rep2",
    "NURF301_RNAseq_Rep1",
    "NURF301_RNAseq_Rep2"
)

scale_facts_RNAseq.dbl <- c(
    1 / 0.696477995,
    1 / 0.645014413,
    1 / 0.75365758,
    1 / 0.596904982,
    1 / 0.744996406,
    1 / 1.00000000,
    1 / 0.707692031,
    1 / 0.490665005,
    1 / 0.668193695,
    1 / 0.772597526
)

scale_facts_RNAseq.lst <-
  setNames(as.list(scale_facts_RNAseq.dbl), samples_RNAseq.chr)

# ATAC-seq DHS (fragments < 120 bp) scaling factors
# (Determined by DESeq2 using counts of DHS signal
# in promoter regions, see below)
samples_ATACDHS.chr <-     c(
    "BAP170_ATACDHS_Rep1",
```

```
    "BAP170_ATACDHS_Rep2",
    "GAF_ATACDHS_Rep1",
    "GAF_ATACDHS_Rep2",
    "LACZ_ATACDHS_Rep1",
    "LACZ_ATACDHS_Rep2",
    "NURF301BAP170_ATACDHS_Rep1",
    "NURF301BAP170_ATACDHS_Rep2",
    "NURF301_ATACDHS_Rep1",
    "NURF301_ATACDHS_Rep2"
)

scale_facts_ATACDHS.dbl <- c(
    1 / 1,
    1 / 1,
    1 / 1,
    1 / 1,
    1 / 1,
    1 / 1,
    1 / 1,
    1 / 1,
    1 / 1,
    1 / 1
)

scale_facts_ATACDHS.lst <-
  setNames(as.list(scale_facts_ATACDHS.dbl), samples_ATACDHS.chr)

# ATAC-seq MN (mononucs, 130 < fragment < 200) scaling factors
# (Determined by DESeq2 using counts of DHS signal
# in promoter regions, see below)
samples_ATACMN.chr <-     c(
    "BAP170_ATACMN_Rep1",
    "BAP170_ATACMN_Rep2",
    "GAF_ATACMN_Rep1",
    "GAF_ATACMN_Rep2",
    "LACZ_ATACMN_Rep1",
    "LACZ_ATACMN_Rep2",
    "NURF301BAP170_ATACMN_Rep1",
    "NURF301BAP170_ATACMN_Rep2",
    "NURF301_ATACMN_Rep1",
    "NURF301_ATACMN_Rep2"
)

scale_facts_ATACMN.dbl <- c(
    1 / 1,
    1 / 1,
    1 / 1,
    1 / 1,
    1 / 1,
    1 / 1,
    1 / 1,
    1 / 1,
    1 / 1,
```

```
    1 / 1
)

scale_facts_ATACMN.lst <-
  setNames(as.list(scale_facts_ATACMN.dbl), samples_ATACMN.chr)
```

Reading in PROseq bw files

```
PROseq.lst <-
  list(
    "BAP170_PROseq_Rep1" = import_bigWig(
      "bw/PROseq/raw_signal/BAP170_PROseq_Rep1_fwd.bw",
      "bw/PROseq/raw_signal/BAP170_PROseq_Rep1_rev.bw"
    ),
    "BAP170_PROseq_Rep2" = import_bigWig(
      "bw/PROseq/raw_signal/BAP170_PROseq_Rep2_fwd.bw",
      "bw/PROseq/raw_signal/BAP170_PROseq_Rep2_rev.bw"
    ),
    "GAF_PROseq_Rep1" = import_bigWig(
      "bw/PROseq/raw_signal/GAF_PROseq_Rep1_fwd.bw",
      "bw/PROseq/raw_signal/GAF_PROseq_Rep1_rev.bw"
    ),
    "GAF_PROseq_Rep2" = import_bigWig(
      "bw/PROseq/raw_signal/GAF_PROseq_Rep2_fwd.bw",
      "bw/PROseq/raw_signal/GAF_PROseq_Rep2_rev.bw"
    ),
    "LACZ_PROseq_Rep1" = import_bigWig(
      "bw/PROseq/raw_signal/LACZ_PROseq_Rep1_fwd.bw",
      "bw/PROseq/raw_signal/LACZ_PROseq_Rep1_rev.bw"
    ),
    "LACZ_PROseq_Rep2" = import_bigWig(
      "bw/PROseq/raw_signal/LACZ_PROseq_Rep2_fwd.bw",
      "bw/PROseq/raw_signal/LACZ_PROseq_Rep2_rev.bw"
    ),
    "NURF301BAP170_PROseq_Rep1" = import_bigWig(
      "bw/PROseq/raw_signal/NURF301BAP170_PROseq_Rep1_fwd.bw",
      "bw/PROseq/raw_signal/NURF301BAP170_PROseq_Rep1_rev.bw"
    ),
    "NURF301BAP170_PROseq_Rep2" = import_bigWig(
      "bw/PROseq/raw_signal/NURF301BAP170_PROseq_Rep2_fwd.bw",
      "bw/PROseq/raw_signal/NURF301BAP170_PROseq_Rep2_rev.bw"
    ),
    "NURF301_PROseq_Rep1" = import_bigWig(
      "bw/PROseq/raw_signal/NURF301_PROseq_Rep1_fwd.bw",
      "bw/PROseq/raw_signal/NURF301_PROseq_Rep1_rev.bw"
    ),
    "NURF301_PROseq_Rep2" = import_bigWig(
      "bw/PROseq/raw_signal/NURF301_PROseq_Rep2_fwd.bw",
      "bw/PROseq/raw_signal/NURF301_PROseq_Rep2_rev.bw"
    )
  )
```

Reading in RNAseq bw files

```
RNAseq.lst <-
  list(
    "BAP170_RNAseq_Rep1" = import_bigWig(
      "bw/RNAseq/raw_signal/BAP_RNAseq_Rep1_fwd.bw",
      "bw/RNAseq/raw_signal/BAP_RNAseq_Rep1_rev.bw"
    ),
    "BAP170_RNAseq_Rep2" = import_bigWig(
      "bw/RNAseq/raw_signal/BAP_RNAseq_Rep2_fwd.bw",
      "bw/RNAseq/raw_signal/BAP_RNAseq_Rep2_rev.bw"
    ),
     "GAF_RNAseq_Rep1" = import_bigWig(
      "bw/RNAseq/raw_signal/GAF_RNAseq_Rep1_fwd.bw",
      "bw/RNAseq/raw_signal/GAF_RNAseq_Rep1_rev.bw"
    ),
    "GAF_RNAseq_Rep2" = import_bigWig(
      "bw/RNAseq/raw_signal/GAF_RNAseq_Rep2_fwd.bw",
      "bw/RNAseq/raw_signal/GAF_RNAseq_Rep2_rev.bw"
    ),
    "LACZ_RNAseq_Rep1" = import_bigWig(
      "bw/RNAseq/raw_signal/LACZ_RNAseq_Rep1_fwd.bw",
      "bw/RNAseq/raw_signal/LACZ_RNAseq_Rep1_rev.bw"
    ),
    "LACZ_RNAseq_Rep2" = import_bigWig(
      "bw/RNAseq/raw_signal/LACZ_RNAseq_Rep2_fwd.bw",
      "bw/RNAseq/raw_signal/LACZ_RNAseq_Rep2_rev.bw"
    ),
    "NURF301_RNAseq_Rep1" = import_bigWig(
      "bw/RNAseq/raw_signal/NURF_RNAseq_Rep1_fwd.bw",
      "bw/RNAseq/raw_signal/NURF_RNAseq_Rep1_rev.bw"
    ),
    "NURF301_RNAseq_Rep2" = import_bigWig(
      "bw/RNAseq/raw_signal/NURF_RNAseq_Rep2_fwd.bw",
      "bw/RNAseq/raw_signal/NURF_RNAseq_Rep2_rev.bw"
    ),
    "NURF301BAP170_RNAseq_Rep1" = import_bigWig(
      "bw/RNAseq/raw_signal/NURFBAP_RNAseq_Rep1_fwd.bw",
      "bw/RNAseq/raw_signal/NURFBAP_RNAseq_Rep1_rev.bw"
    ),
    "NURF301BAP170_RNAseq_Rep2" = import_bigWig(
      "bw/RNAseq/raw_signal/NURFBAP_RNAseq_Rep2_fwd.bw",
      "bw/RNAseq/raw_signal/NURFBAP_RNAseq_Rep2_rev.bw"
    )
  )
```

Reading in ATAC-seq bw files

```
# First DHS, which is ATAC-seq that's been filtered for fragments < 120 bp
# only (entire fragments piled up for signal)
ATACDHS.lst <-
```

```
  list(
    "BAP170_ATACDHS_Rep1" =
      tidyChromosomes(
        import.bw("bw/ATACseq/DHS/raw_signal/BAP170_Rep1_merged_AtacDHS.bw")
      ),
    "BAP170_ATACDHS_Rep2" =
      tidyChromosomes(
        import.bw("bw/ATACseq/DHS/raw_signal/BAP170_Rep2_merged_AtacDHS.bw")
      ),
    "GAF_ATACDHS_Rep1" =
      tidyChromosomes(
        import.bw("bw/ATACseq/DHS/raw_signal/GAF_Rep1_merged_AtacDHS.bw")
      ),
    "GAF_ATACDHS_Rep2" =
      tidyChromosomes(
        import.bw("bw/ATACseq/DHS/raw_signal/GAF_Rep2_merged_AtacDHS.bw")
      ),
    "LACZ_ATACDHS_Rep1" =
      tidyChromosomes(
        import.bw("bw/ATACseq/DHS/raw_signal/LACZ_Rep1_merged_AtacDHS.bw")
      ),
    "LACZ_ATACDHS_Rep2" =
      tidyChromosomes(
        import.bw("bw/ATACseq/DHS/raw_signal/LACZ_Rep2_merged_AtacDHS.bw")
      ),
    "NURF301_ATACDHS_Rep1" =
      tidyChromosomes(
        import.bw("bw/ATACseq/DHS/raw_signal/NURF301_Rep1_merged_AtacDHS.bw")
      ),
    "NURF301_ATACDHS_Rep2" =
      tidyChromosomes(
        import.bw("bw/ATACseq/DHS/raw_signal/NURF301_Rep2_merged_AtacDHS.bw")
      ),
    "NURF301BAP170_ATACDHS_Rep1" =
      tidyChromosomes(
        import.bw(
          "bw/ATACseq/DHS/raw_signal/NURF301BAP170_Rep1_merged_AtacDHS.bw"
        )
      ),
    "NURF301BAP170_ATACDHS_Rep2" =
      tidyChromosomes(
        import.bw(
          "bw/ATACseq/DHS/raw_signal/NURF301BAP170_Rep2_merged_AtacDHS.bw"
        )
      )
  )

# Now MN, which is ATAC-seq thats 130 bp < fragment < 200 bp
# (only central 3 bp contribute to signal)

ATACMN.lst <-
  list(
    "BAP170_ATACMN_Rep1" =
```

```
    tidyChromosomes(
      import.bw("bw/ATACseq/MN/raw_signal/BAP170_Rep1_merged_AtacMN.bw")
    ),
  "BAP170_ATACMN_Rep2" =
    tidyChromosomes(
      import.bw("bw/ATACseq/MN/raw_signal/BAP170_Rep2_merged_AtacMN.bw")
    ),
  "GAF_ATACMN_Rep1" =
    tidyChromosomes(
      import.bw("bw/ATACseq/MN/raw_signal/GAF_Rep1_merged_AtacMN.bw")
    ),
  "GAF_ATACMN_Rep2" =
    tidyChromosomes(
      import.bw("bw/ATACseq/MN/raw_signal/GAF_Rep2_merged_AtacMN.bw")
    ),
  "LACZ_ATACMN_Rep1" =
    tidyChromosomes(
      import.bw("bw/ATACseq/MN/raw_signal/LACZ_Rep1_merged_AtacMN.bw")
    ),
  "LACZ_ATACMN_Rep2" =
    tidyChromosomes(
      import.bw("bw/ATACseq/MN/raw_signal/LACZ_Rep2_merged_AtacMN.bw")
    ),
  "NURF301_ATACMN_Rep1" =
    tidyChromosomes(
      import.bw("bw/ATACseq/MN/raw_signal/NURF301_Rep1_merged_AtacMN.bw")
    ),
  "NURF301_ATACMN_Rep2" =
    tidyChromosomes(
      import.bw("bw/ATACseq/MN/raw_signal/NURF301_Rep2_merged_AtacMN.bw")
    ),
  "NURF301BAP170_ATACMN_Rep1" =
    tidyChromosomes(
      import.bw(
        "bw/ATACseq/MN/raw_signal/NURF301BAP170_Rep1_merged_AtacMN.bw"
      )
    ),
  "NURF301BAP170_ATACMN_Rep2" =
    tidyChromosomes(
      import.bw(
        "bw/ATACseq/MN/raw_signal/NURF301BAP170_Rep2_merged_AtacMN.bw"
      )
    )
)
```

Reading in ChIP/CUT&RUN bw files

```
# Reading in list of CUT&RUN bigWigs
CaR.lst <-
  list(
    "GAF_CUTandRUN" = tidyChromosomes(
      import.bw("bw/CutAndRun/GAF_CUTandRUN.bw")
```

```
      ),
    "NURF_CUTandRUN" = tidyChromosomes(
      import.bw("bw/CutAndRun/NURF_CUTandRUN.bw")
      )
    )

# RPM normalizing CUT&RUN signal
CaR.lst <- lapply(CaR.lst, RPMnorm)

# Reading in list of ChIP-seq bigWigs
ChIP.lst <-
    list(
        "GAF_ChIPseq" = tidyChromosomes(
          import.bw("bw/ChIPseq/GAF_ChIPseq.bw")
          ),
        "SP1_ChIPseq" = tidyChromosomes(
          import.bw("bw/ChIPseq/SP1_ChIPseq.bw")
          ),
        "M1BP_ChIPseq" = tidyChromosomes(
          import.bw("bw/ChIPseq/M1BP_ChIPseq.bw")
          ),
        "BEAF32_ChIPseq" = tidyChromosomes(
          import.bw("bw/ChIPseq/BEAF32_ChIPseq.bw")
          )
    )

# RPM normalizing ChIP-seq signal
ChIP.lst <- lapply(ChIP.lst, RPMnorm)
```

Generating count tables

```
# Gene body (TSS+200 to TES-200)
# initialize matrix
GB.tbl <- tibble("tx_name" = genes.tbl$tx_name)
# populate with counts
for(i in samples_PROseq.chr){
  GB.tbl[i] <- getCountsByRegions(
    PROseq.lst[[i]], genebodies(genes.gr, 200, -200))
}


## Promoter (TSS-50 to TSS+100)
PR.tbl <- tibble("tx_name" = genes.tbl$tx_name)
for(i in samples_PROseq.chr){
  PR.tbl[i] <- getCountsByRegions(
    PROseq.lst[[i]], genebodies(genes.gr, -50, 100, fix.end = "start"))
}

## Upstream (TSS-500 to TSS-200)
US.tbl <- tibble("tx_name" = genes.tbl$tx_name)
for(i in samples_PROseq.chr){
```

```
  US.tbl[i] <- getCountsByRegions(
    PROseq.lst[[i]], genebodies(genes.gr, -500, -200, fix.end = "start"))
}

# Getting RNA-seq count data
RNA.tbl <- tibble("tx_name" = genes.tbl$tx_name)
for(i in samples_RNAseq.chr){
  RNA.tbl[i] <- getCountsByRegions(
    RNAseq.lst[[i]], genebodies(genes.gr, -1000, 0, fix.start = "end"))
}

# Getting ATAC-seq counts in peaks
ATAC_peaks.tbl <- tibble("peak_name" = ATACseq_summits.gr$peak_name)
for(i in samples_ATACDHS.chr){
  ATAC_peaks.tbl[i] <-
    getCountsByRegions(
      ATACDHS.lst[[i]],
      promoters(ATACseq_summits.gr, 100, 100),
      expand_ranges = TRUE)
}

# Getting ATAC-seq counts at promoters
ATAC_PR.tbl <- tibble("tx_name" = genes.gr$tx_name)
for(i in samples_ATACDHS.chr){
  ATAC_PR.tbl[i] <-
    getCountsByRegions(
      ATACDHS.lst[[i]],
      promoters(genes.gr, 1000, 0),
      expand_ranges = TRUE)
}

# Getting ATAC-seq counts (mononucleosome sized fragments)
# in the first 1.5kb of each gene
ATACMN_genes.tbl <- tibble("tx_name" = genes.gr$tx_name)
for(i in samples_ATACMN.chr){
  ATACMN_genes.tbl[i] <-
    getCountsByRegions(
      ATACMN.lst[[i]], promoters(genes.gr, 0, 1500), expand_ranges = TRUE)
}
```

Blacklisting genes with too much upstream transcription

If an upstream gene is differentially expressed in one condition, it can appear like a downstream gene has differentially expressed pausing due to read-through transcription (see Duarte et al. Genes Dev 2016. doi:10.1101/gad.284430.116 for more detailed explanation). To eliminate this problem, we filter out any genes in our gene list that have more than half the PRO-seq signal in the upstream region (TSS-500 to TSS-200) as is observed in the pause region (TSS-50 to TSS+100), or genes which have more signal in the upstream region than in the length normalized gene body. This results in blacklisting 5099 transcripts, or ~35% of annotated unique transcripts over 500 bp, leaving 9375 genes for further analysis

```
# Normalizing data using spike-in scale factors
# Gene body (TSS+200 to TES-200)
GB_normed.tbl <- GB.tbl
```

```r
for (i in colnames(GB.tbl[, -1])) {
    GB_normed.tbl[, i] <- GB.tbl[, i] / scale_facts_PROseq.lst[[i]]
}

# Promoter (TSS-50 to TSS+100)
PR_normed.tbl <- PR.tbl
for (i in colnames(PR.tbl[, -1])) {
    PR_normed.tbl[, i] <- PR.tbl[, i] / scale_facts_PROseq.lst[[i]]
}

# Upstream (TSS-500 to TSS-200)
US_normed.tbl <- US.tbl
for (i in colnames(US.tbl[, -1])) {
    US_normed.tbl[, i] <- US.tbl[, i] / scale_facts_PROseq.lst[[i]]
}

# Initializing data frame of replicate averages of
# LACZ for PR, GB, & US for each transcript. Also
# getting gene length to length matched normalize
# GB signal
blacklist.tbl <- data.frame(
    "tx_name" = GB_normed.tbl$tx_name,
    "gene_length" = genes.tbl$stop - genes.tbl$start,
    "GB" = apply(
      GB_normed.tbl[, c("LACZ_PROseq_Rep1", "LACZ_PROseq_Rep2")], 1, mean
      ),
    "PR" = apply(
      PR_normed.tbl[, c("LACZ_PROseq_Rep1", "LACZ_PROseq_Rep2")], 1, mean
      ),
    "US" = apply(
      US_normed.tbl[, c("LACZ_PROseq_Rep1", "LACZ_PROseq_Rep2")], 1, mean
      )
)

# Length normalizing GB to RPK
blacklist.tbl$GB <-
    blacklist.tbl$GB * (1000 / blacklist.tbl$gene_length)

# Length normalizing PR to RPK
blacklist.tbl$PR <-
    blacklist.tbl$PR * (1000 / 150)

# Length normalizing US to RPK
blacklist.tbl$US <-
    blacklist.tbl$US * (1000 / 300)

# Blacklisting genes with US transcription equal to more than 0.5X PR or GB
blacklist.chr <- (unique(rbind(
    blacklist.tbl[which(blacklist.tbl$US > (0.5 * blacklist.tbl$PR)),],
    blacklist.tbl[which(blacklist.tbl$US > blacklist.tbl$GB),])
))$tx_name
```

```
# Filtering blacklisted genes out of all data loaded so far
GB.tbl <-
  GB.tbl[which(!(GB.tbl$tx_name %in% blacklist.chr)),]
PR.tbl <-
  PR.tbl[which(!(PR.tbl$tx_name %in% blacklist.chr)),]
GB_normed.tbl <-
  GB_normed.tbl[which(!(GB_normed.tbl$tx_name %in% blacklist.chr)),]
PR_normed.tbl <-
  PR_normed.tbl[which(!(PR_normed.tbl$tx_name %in% blacklist.chr)),]
genes.tbl <-
  genes.tbl[which(!(genes.tbl$tx_name %in% blacklist.chr)),]
genes.gr <-
  genes.gr[which(!(genes.gr$tx_name %in% blacklist.chr)),]
ATAC_PR.tbl <-
  ATAC_PR.tbl[which(!(ATAC_PR.tbl$tx_name %in% blacklist.chr)),]
ATACMN_genes.tbl <-
  ATACMN_genes.tbl[which(!(ATACMN_genes.tbl$tx_name %in% blacklist.chr)),]

# Saving gene list for loading later
write_tsv(genes.tbl[,-c(7, 8)], path = "bed/filtered_dm6_genes.bed", col_names = FALSE)
```

Normalizing, pooling replicates, and saving bw files

```
# Normalizing PRO-seq in place using scale factors
for (i in samples_PROseq.chr) {
   PROseq.lst[[i]]$score <- PROseq.lst[[i]]$score / scale_facts_PROseq.lst[[i]]
}

# Merging replicates (summing them across bins) and saving
PROseq_normed_merged.lst <- vector(mode = "list", length = 0L)
for(i in samples_PROseq.chr[
  which(unlist(lapply(
    samples_PROseq.chr, grepl, pattern = "Rep1"
    )))]
) {
  PROseq_normed_merged.lst[[gsub("_Rep1", "", i)]] <-
    mergeGRangesData(
      PROseq.lst[[i]],
      PROseq.lst[[gsub("_Rep1", "_Rep2", i)]]
      )
}

# Saving bw files for future loading
for(i in names(PROseq_normed_merged.lst)){
  export.bw(
    PROseq_normed_merged.lst[[i]][which(
      strand(PROseq_normed_merged.lst[[i]]) == "+")],
    paste0("bw/PROseq/merged_normed/", gsub("seq", "seq_fwd.bw", i))
  )
  export.bw(
    PROseq_normed_merged.lst[[i]][which(
```

```
      strand(PROseq_normed_merged.lst[[i]]) == "-")],
    paste0("bw/PROseq/merged_normed/", gsub("seq", "seq_rev.bw", i))
  )
}

# Normalizing RNA-seq in place using scale factors
for (i in samples_RNAseq.chr) {
   RNAseq.lst[[i]]$score <-
     RNAseq.lst[[i]]$score / scale_facts_RNAseq.lst[[i]]
}

# Merging replicates (summing them across bins) and saving
RNAseq_normed_merged.lst <- vector(mode = "list", length = 0L)
for(i in samples_RNAseq.chr[
  which(unlist(lapply(
    samples_RNAseq.chr, grepl, pattern = "Rep1"
    )))]
) {
  RNAseq_normed_merged.lst[[gsub("_Rep1", "", i)]] <-
    mergeGRangesData(
      RNAseq.lst[[i]],
      RNAseq.lst[[gsub("_Rep1", "_Rep2", i)]]
      )
}

# Saving bw files for future loading
for(i in names(RNAseq_normed_merged.lst)){
  export.bw(
    RNAseq_normed_merged.lst[[i]][which(
      strand(RNAseq_normed_merged.lst[[i]]) == "+")],
    paste0("bw/RNAseq/merged_normed/", gsub("seq", "seq_fwd.bw", i))
  )
  export.bw(
    RNAseq_normed_merged.lst[[i]][which(
      strand(RNAseq_normed_merged.lst[[i]]) == "-")],
    paste0("bw/RNAseq/merged_normed/", gsub("seq", "seq_rev.bw", i))
  )
}

# Normalizing ATACDHS in place using scale factors
for (i in samples_ATACDHS.chr) {
   ATACDHS.lst[[i]]$score <-
     ATACDHS.lst[[i]]$score / scale_facts_ATACDHS.lst[[i]]
}

# Merging replicates (summing them across bins) and saving
BAP170_merged.gr <-
  mergeGRangesData(
    ATACDHS.lst["BAP170_ATACDHS_Rep1"],
    ATACDHS.lst["BAP170_ATACDHS_Rep2"])
export.bw(
  BAP170_merged.gr,
  "bw/ATACseq/DHS/merged_normed/BAP170_ATACDHS.bw")
```

```
GAF_merged.gr <-
  mergeGRangesData(
    ATACDHS.lst["GAF_ATACDHS_Rep1"],
    ATACDHS.lst["GAF_ATACDHS_Rep2"])
export.bw(
  GAF_merged.gr,
  "bw/ATACseq/DHS/merged_normed/GAF_ATACDHS.bw")

LACZ_merged.gr <-
  mergeGRangesData(
    ATACDHS.lst["LACZ_ATACDHS_Rep1"],
    ATACDHS.lst["LACZ_ATACDHS_Rep2"])
export.bw(
  LACZ_merged.gr,
  "bw/ATACseq/DHS/merged_normed/LACZ_ATACDHS.bw")

NURF301_merged.gr <-
  mergeGRangesData(
    ATACDHS.lst["NURF301_ATACDHS_Rep1"],
    ATACDHS.lst["NURF301_ATACDHS_Rep2"])
export.bw(
  NURF301_merged.gr,
  "bw/ATACseq/DHS/merged_normed/NURF301_ATACDHS.bw")

NURF301BAP170_merged.gr <-
  mergeGRangesData(
    ATACDHS.lst["NURF301BAP170_ATACDHS_Rep1"],
    ATACDHS.lst["NURF301BAP170_ATACDHS_Rep2"])
export.bw(
  NURF301BAP170_merged.gr,
  "bw/ATACseq/DHS/merged_normed/NURF301BAP170_ATACDHS.bw")


# Getting DESeq Normalization factors for ATACMN data
# Getting counts matrix
ATACMN_genes.df <-
    data.frame(ATACMN_genes.tbl, row.names = "tx_name")

# colData object for DESeq2
col_data.df <- data.frame(
    row.names = colnames(ATACMN_genes.df),
    RNAi = c(
        "BAP170",
        "BAP170",
        "GAF",
        "GAF",
        "LACZ",
        "LACZ",
        "NURF301BAP170",
        "NURF301BAP170",
        "NURF301",
        "NURF301"
    )
```

```
)

# construction of DESeq2 object
ATACMN_genes.dds <-
    DESeqDataSetFromMatrix(countData = ATACMN_genes.df,
                           colData = col_data.df,
                           design = ~ RNAi)


# Running DESeq2
ATACMN_genes.dds <- DESeq(ATACMN_genes.dds)

scale_facts_ATACMN.dbl <- sizeFactors(ATACMN_genes.dds)
scale_facts_ATACMN.lst <-
    setNames(as.list(scale_facts_ATACMN.dbl), samples_ATACMN.chr)

# Normalizing ATACMN in place using scale factors
for (i in samples_ATACMN.chr) {
    ATACMN.lst[[i]]$score <-
        ATACMN.lst[[i]]$score / scale_facts_ATACMN.lst[[i]]
}

# Merging replicates (summing them across bins) and saving
BAP170_merged.gr <-
    mergeGRangesData(
      ATACMN.lst["BAP170_ATACMN_Rep1"],
      ATACMN.lst["BAP170_ATACMN_Rep2"])
export.bw(
  BAP170_merged.gr,
  "bw/ATACseq/MN/merged_normed/BAP170_ATACMN.bw")

GAF_merged.gr <-
    mergeGRangesData(
      ATACMN.lst["GAF_ATACMN_Rep1"],
      ATACMN.lst["GAF_ATACMN_Rep2"])
export.bw(
  GAF_merged.gr,
  "bw/ATACseq/MN/merged_normed/GAF_ATACMN.bw")

LACZ_merged.gr <-
    mergeGRangesData(
      ATACMN.lst["LACZ_ATACMN_Rep1"],
      ATACMN.lst["LACZ_ATACMN_Rep2"])
export.bw(
  LACZ_merged.gr,
  "bw/ATACseq/MN/merged_normed/LACZ_ATACMN.bw")

NURF301_merged.gr <-
    mergeGRangesData(
      ATACMN.lst["NURF301_ATACMN_Rep1"],
      ATACMN.lst["NURF301_ATACMN_Rep2"])
export.bw(
  NURF301_merged.gr,
```

```
  "bw/ATACseq/MN/merged_normed/NURF301_ATACMN.bw")

NURF301BAP170_merged.gr <-
    mergeGRangesData(
      ATACMN.lst["NURF301BAP170_ATACMN_Rep1"],
      ATACMN.lst["NURF301BAP170_ATACMN_Rep2"])
export.bw(
  NURF301BAP170_merged.gr,
  "bw/ATACseq/MN/merged_normed/NURF301BAP170_ATACMN.bw")
```

Reading back in normalized data

```
rm(ATACDHS.lst)
rm(PROseq.lst)
rm(RNAseq.lst)
rm(ATACMN.lst)

# ATACMN
ATACMN_normed_merged.lst <-
  list(
    "LACZ_ATACMN" =
        tidyChromosomes(
          import.bw("bw/ATACseq/MN/merged_normed/LACZ_ATACMN.bw")
          ),
    "GAF_ATACMN" =
        tidyChromosomes(
          import.bw("bw/ATACseq/MN/merged_normed/GAF_ATACMN.bw")
          ),
    "BAP170_ATACMN" =
        tidyChromosomes(
          import.bw("bw/ATACseq/MN/merged_normed/BAP170_ATACMN.bw")
          ),
    "NURF301_ATACMN" =
        tidyChromosomes(
          import.bw("bw/ATACseq/MN/merged_normed/NURF301_ATACMN.bw")
          ),
    "NURF301BAP170_ATACMN" =
        tidyChromosomes(
          import.bw("bw/ATACseq/MN/merged_normed/NURF301BAP170_ATACMN.bw")
          )
  )

# ATACDHS
ATACDHS_normed_merged.lst <-
  list(
    "LACZ_ATACDHS" =
        tidyChromosomes(
          import.bw("bw/ATACseq/DHS/merged_normed/LACZ_ATACDHS.bw")
          ),
    "GAF_ATACDHS" =
        tidyChromosomes(
```

```
          import.bw("bw/ATACseq/DHS/merged_normed/GAF_ATACDHS.bw")
          ),
    "BAP170_ATACDHS" =
        tidyChromosomes(
          import.bw("bw/ATACseq/DHS/merged_normed/BAP170_ATACDHS.bw")
          ),
    "NURF301_ATACDHS" =
        tidyChromosomes(
          import.bw("bw/ATACseq/DHS/merged_normed/NURF301_ATACDHS.bw")),
    "NURF301BAP170_ATACDHS" =
        tidyChromosomes(
          import.bw("bw/ATACseq/DHS/merged_normed/NURF301BAP170_ATACDHS.bw")
          )
  )

# PROseq
PROseq_normed_merged.lst <-
    list(
        "LACZ_PROseq" = import_bigWig(
            "bw/PROseq/merged_normed/LACZ_PROseq_fwd.bw",
            "bw/PROseq/merged_normed/LACZ_PROseq_rev.bw"
        ),
        "GAF_PROseq" = import_bigWig(
            "bw/PROseq/merged_normed/GAF_PROseq_fwd.bw",
            "bw/PROseq/merged_normed/GAF_PROseq_rev.bw"
        ),
        "BAP170_PROseq" = import_bigWig(
            "bw/PROseq/merged_normed/BAP170_PROseq_fwd.bw",
            "bw/PROseq/merged_normed/BAP170_PROseq_rev.bw"
        ),
        "NURF301_PROseq" = import_bigWig(
            "bw/PROseq/merged_normed/NURF301_PROseq_fwd.bw",
            "bw/PROseq/merged_normed/NURF301_PROseq_rev.bw"
        ),
        "NURF301BAP170_PROseq" = import_bigWig(
            "bw/PROseq/merged_normed/NURF301BAP170_PROseq_fwd.bw",
            "bw/PROseq/merged_normed/NURF301BAP170_PROseq_rev.bw"
        )
    )

# RNAseq
RNAseq_normed_merged.lst<-
    list(
        "LACZ_RNAseq" = import_bigWig(
            "bw/RNAseq/merged_normed/LACZ_RNAseq_fwd.bw",
            "bw/RNAseq/merged_normed/LACZ_RNAseq_rev.bw"
        ),
        "GAF_RNAseq" = import_bigWig(
            "bw/RNAseq/merged_normed/GAF_RNAseq_fwd.bw",
            "bw/RNAseq/merged_normed/GAF_RNAseq_rev.bw"
        ),
        "BAP170_RNAseq" = import_bigWig(
            "bw/RNAseq/merged_normed/BAP170_RNAseq_fwd.bw",
```

```
            "bw/RNAseq/merged_normed/BAP170_RNAseq_rev.bw"
        ),
        "NURF301_RNAseq" = import_bigWig(
            "bw/RNAseq/merged_normed/NURF301_RNAseq_fwd.bw",
            "bw/RNAseq/merged_normed/NURF301_RNAseq_rev.bw"
        ),
        "NURF301BAP170_RNAseq" = import_bigWig(
            "bw/RNAseq/merged_normed/NURF301BAP170_RNAseq_fwd.bw",
            "bw/RNAseq/merged_normed/NURF301BAP170_RNAseq_rev.bw"
        )
    )

genes.gr <- makeGRangesFromDataFrame(
    read_tsv("bed/filtered_dm6_genes.bed",
             col_names = c(
                 "chr", "start", "end",
                 "gene_name", "tx_name",
                 "strand"
             )),
    keep.extra.columns = T
)

# Reading in M1BP PRO-seq
PROseq_M1BP.lst <- list(
  "LACZ" = import_bigWig(
    "bw/M1BP_PROseq/LACZ_fwd.bw",
    "bw/M1BP_PROseq/LACZ_rev.bw"
  ),
  "M1BP" = import_bigWig(
    "bw/M1BP_PROseq/M1BP_fwd.bw",
    "bw/M1BP_PROseq/M1BP_rev.bw"
  )
)
```

Figure 1

A - Overview cartoon (not made in R, inserted in Illustrator later)
B - PCA of promoter PRO-seq
C - glob1 browser shot
D - GAF-dependent promoters ATAC-seq (<120bp) metaprofile
E - GAF-dependent promoters PRO-seq (<120bp) metaprofile
F - Duplicate of E - easier to remove irrelevant lines from each later
G - GAF PR l2FC vs PBAP and NURF scatterplot

```
# Panel A - just use plot spacer and insert cartoon in Illustrator later

# Panel B - PCA of promoter PRO-seq
# Formatting count matrix for DESeq2.
# Rownames = gene names, only count data
PR.df <- data.frame(PR.tbl, row.names = "tx_name")

# colData object for DESeq2
col_data.df <- data.frame(row.names = colnames(PR.df),
```

```
                      RNAi = c("BAP170","BAP170",
                               "GAF", "GAF",
                               "LACZ", "LACZ",
                               "NURF301BAP170", "NURF301BAP170",
                               "NURF301", "NURF301"))

# Construction of DESeq2 object
PR.dds <- DESeqDataSetFromMatrix(countData = PR.df,
                                 colData = col_data.df,
                                 design = ~ RNAi)

# Adding spike-in scale factors
sizeFactors(PR.dds) <- scale_facts_PROseq.dbl

# Running DESeq2
PR.dds <- DESeq(PR.dds)

# Variance stabilizing log transform
PR.rld <- rlog(PR.dds, blind = F)

# Getting PCA data
PR.pca <-
   plotPCA(PR.rld,
           intgroup = c("RNAi"),
           returnData = TRUE)
# Getting % Var explained by each PC
PR_pcavar.dbl <- round(100 * attr(PR.pca, "percentVar"))

# Factoring for plotting
PR.pca$RNAi <-
    factor(PR.pca$RNAi,
           levels = c("LACZ", "GAF", "BAP170", "NURF301", "NURF301BAP170"))

# Plotting
F1B <- ggplot(PR.pca, aes(PC1, PC2, color = RNAi)) +
    geom_point(size = 1.5) +
    xlab(paste0("PC1: ", PR_pcavar.dbl[1], "% var.")) +
    ylab(paste0("PC2: ", PR_pcavar.dbl[2], "% var.")) +
    ggtheme.jj() +
    scale_color_manual(values = RNAi_cols.chr,
                       breaks = c("LACZ", "GAF", "BAP170", "NURF301", "NURF301BAP170"),
                       labels = c("LACZ", "GAF", "BAP", "NURF", "N+B"))+
    scale_x_continuous(limits = c(-50, 50),
                       expand = c(0, 0),
                       breaks = c(-50, -25, 0, 25, 50))+
        scale_y_continuous(limits = c(-30, 30),
                       expand = c(0,0),
                       breaks = c(-30,-15, 0, 15, 30))

# Panel C - glob1 browser shot
# Getting granges of gene
glob1.gr <- genes.gr[which(genes.gr$tx_name == "glob1-RB")]
```

```r
# Moving end of gene in (to condense plot)
start(glob1.gr) <- start(glob1.gr) + 2000

# Plotting (see file browserPlotR.R for more detail)
F1C <- browser_plotter.jj(
    glob1.gr,
    list("PROseq" = PROseq_normed_merged.lst,
         "ATACDHS" = ATACDHS_normed_merged.lst,
         "ChIPseq" = ChIP.lst["GAF_ChIPseq"]),
    labs = c("LACZ", "GAF", "BAP", "NURF", "N+B",
             "LACZ", "GAF", "BAP", "NURF", "N+B",
             "GAF ChIP"),
    scale_bar_size = 1000,
    pad_left = 0,
    pad_right = 500,
    binsize = 1,
    bin_FUN = mean,
    .expand_ranges = list(
        "PROseq" = FALSE,
        "ATACDHS" = TRUE,
        "ChIPseq" = TRUE)
    )

# Panel D - GAF-dependent promoters ATAC-seq (<120bp) metaprofile
# Get df of counts for running DESeq2
PR.df <- data.frame(PR.tbl, row.names = "tx_name")
PR_col_data.df <- data.frame(
  row.names = colnames(PR.df),
  RNAi = c(
    "BAP170",
    "BAP170",
    "GAF",
    "GAF",
    "LACZ",
    "LACZ",
    "NURFBAP",
    "NURFBAP",
    "NURF301",
    "NURF301"
  )
)

# Getting DESeq results object, GAF vs LACZ
PR_GAF.res <-
  subset_DESeq.jj(
    PR.df,
    PR_col_data.df,
    scale_facts_PROseq.dbl,
    c(3:6),
    c("RNAi", "GAF", "LACZ"),
    ~RNAi)

# Converting results object to df
```

```r
PR_GAFres.df <- as.data.frame(PR_GAF.res)
PR_GAFres.df$tx_name <- row.names(PR_GAFres.df)

# Writing results object as file
write_tsv(PR_GAFres.df, "DESeq_results/GAF_PR_res.tsv")

# Getting list of GAF-dependent promoters (FDR 0.01)
gafDepPR.gr <-
    genes.gr[which(
        genes.gr$tx_name %in%
            (PR_GAFres.df %>%
            filter(padj < 0.01 & log2FoldChange < 0))$tx_name
    )]

# Getting counts matrix of ATAC data
gafDepPR_ATACDHSSub.mat <-
    metaSubsample(
        dataset.gr = ATACDHS_normed_merged.lst,
        regions.gr = promoters(gafDepPR.gr, 500, 500),
        binsize = 1,
        first.output.xval = -500,
        expand_ranges = TRUE
    )

# Factoring sample.name column
gafDepPR_ATACDHSSub.mat$sample.name <-
    factor(
        gafDepPR_ATACDHSSub.mat$sample.name,
        levels = names(ATACDHS_normed_merged.lst)
    )

# Plotting
F1D <- ggMetaplot(gafDepPR_ATACDHSSub.mat) +
    gg_color_scale_RNAi_ATACDHS +
    gg_fill_scale_RNAi_ATACDHS +
    scale_x_continuous(
        limits = c(-500, 500),
        expand = c(0,0)
    ) +
    scale_y_continuous(
        limits = c(0, 200),
        expand = c(0,0)
    )+
  ggtitle("ATAC < 120 bp")
# Panel E - GAF-dependent promoters PRO-seq (<120bp) metaprofile
# Getting subsampled count matrices
gafDepPR_proSeqSub.mat <-
     metaSubsample(
        dataset.gr = PROseq_normed_merged.lst,
        regions.gr = promoters(gafDepPR.gr, 0, 150),
        binsize = 2,
        first.output.xval = 0,
        expand_ranges = FALSE
```

```
    )


gafDepGB_proSeqSub.mat <-
     metaSubsample(
        dataset.gr = PROseq_normed_merged.lst,
        regions.gr = promoters(gafDepPR.gr, 0, 1500),
        binsize = 20,
        first.output.xval = 0,
        expand_ranges = FALSE
    )

# Filling sample.name column and factoring
gafDepPR_proSeqSub.mat$sample.name <-
    factor(
        gafDepPR_proSeqSub.mat$sample.name,
        levels = names(PROseq_normed_merged.lst)
    )

gafDepGB_proSeqSub.mat$sample.name <-
    factor(
        gafDepGB_proSeqSub.mat$sample.name,
        levels = names(PROseq_normed_merged.lst)
    )


# Plotting promoter
F1EFa <- ggMetaplot(gafDepPR_proSeqSub.mat)+
    gg_color_scale_RNAi_PROseq +
    gg_fill_scale_RNAi_PROseq +
    scale_x_continuous(
        limits = c(0, 150),
        expand = c(0,0),
        breaks = c(0, 50, 100, 150)
    ) +
    scale_y_continuous(
        breaks = c(0, 50, 100)
    ) +
    coord_cartesian(
        ylim = c(0, 100),
        expand = 0,
        )+
  ggtitle("PRO-seq")


# Plotting gene body
F1EFb <- ggMetaplot(gafDepGB_proSeqSub.mat)+
    gg_color_scale_RNAi_PROseq +
    gg_fill_scale_RNAi_PROseq +
    scale_y_continuous(
        breaks = c(0, 0.25, 0.5, 0.75)
    )+
    scale_x_continuous(
```

```
        breaks = c(seq(150, 1500, length.out = 4))
    )+
    coord_cartesian(
        xlim = c(150, 1500),
        ylim = c(0, 0.75),
        expand = 0,
        ) +
    ylab(NULL)

# Panel F - Duplicate of E - easier to remove irrelevant lines from each later

# Panel G - GAF PR l2FC vs PBAP and NURF scatterplot
# Running DESeq2 for BAP and NURF vs LACZ control
PR_BAP.res <-
  subset_DESeq.jj(
    PR.df,
    PR_col_data.df,
    scale_facts_PROseq.dbl,
    c(1, 2, 5, 6),
    c("RNAi", "BAP170", "LACZ"),
    ~ RNAi
  )

PR_NURF.res <-
  subset_DESeq.jj(
    PR.df,
    PR_col_data.df,
    scale_facts_PROseq.dbl,
    c(5, 6, 9, 10),
    c("RNAi", "NURF301", "LACZ"),
    ~ RNAi
  )


# Plotting GAF vs BAP
F1Ga <- fc_corr.jj(PR_GAF.res, PR_BAP.res)+
  coord_cartesian(xlim = c(-8, 8), ylim = c(-8, 8), expand = FALSE)+
  xlab("GAF PR l2FC")+
  ylab("BAP170 PR l2FC")

# Plotting GAF vs NURF
F1Gb <- fc_corr.jj(PR_GAF.res, PR_NURF.res)+
  coord_cartesian(xlim = c(-8, 8), ylim = c(-8, 8), expand = FALSE)+
  xlab("GAF PR l2FC")+
  ylab("BAP170 PR l2FC")

# Layout for plotting
F1_layout <- "
#C#
ADE
BFG
BHI
"
```

```
# Combining all panels
F1_all <- F1B + F1C + F1D + F1EFa +
  F1EFb + F1EFa + F1EFb + F1Ga + F1Gb +
    plot_layout(guides = "collect",
                design = F1_layout) +
    plot_annotation(tag_levels = 'A') &
    theme(plot.tag = element_text(family = "Helvetica", face = "bold", size = 12))
```

Figure S1

A - RNA-seq knockdown efficiency
B/C - Western blots showing knockdown efficiency of various factors. No analysis needed, assembled in Illustrator

```
# Panel A - Knockdown efficiency by RNA-seq

# Normalizing RNA-seq counts using ERCC counts
# scale_factor = minimum(mapped_ERCC) / sample(mapped_ERCC)
RNA_normed.tbl <- RNA.tbl
for (i in colnames(RNA.tbl[, -1])) {
    RNA_normed.tbl[, i] <- RNA.tbl[, i] / scale_facts_RNAseq.lst[[i]]
}

# Filtering just rows for the three factors that were knocked down
# and dividing each by their transcript abundance in the lacZ condition
# to get a fold change value
KD.tbl <-
    RNA_normed.tbl[
        which(
          RNA_normed.tbl$tx_name %in% c("Trl-RA", "Bap170-RA", "E(bx)-RC")
          ),
        ] %>%
    transmute(
        "BAP_Rep1_fc" = (BAP170_RNAseq_Rep1 / LACZ_RNAseq_Rep1),
        "BAP_Rep2_fc" = (BAP170_RNAseq_Rep2 / LACZ_RNAseq_Rep2),
        "GAF_Rep1_fc" = (GAF_RNAseq_Rep1 / LACZ_RNAseq_Rep1),
        "GAF_Rep2_fc" = (GAF_RNAseq_Rep2 / LACZ_RNAseq_Rep2),
        "NURFBAP_Rep1_fc" = (NURF301BAP170_RNAseq_Rep1 / LACZ_RNAseq_Rep1),
        "NURFBAP_Rep2_fc" = (NURF301BAP170_RNAseq_Rep2 / LACZ_RNAseq_Rep2),
        "NURF_Rep1_fc" = (NURF301_RNAseq_Rep1 / LACZ_RNAseq_Rep1),
        "NURF_Rep2_fc" = (NURF301_RNAseq_Rep2 / LACZ_RNAseq_Rep2),
        "tx_name" = tx_name
    ) %>%
    mutate("tx_name" =
               if_else(tx_name == "Trl-RA", "GAF",
                       if_else(tx_name == "Bap170-RA", "BAP",
                               if_else(tx_name == "E(bx)-RC", "NURF",
                                       "no_match_found")))) %>%
    gather("target", "FC", -tx_name) %>%
    separate(target, into = c("target", NA, NA))
```

```r
# Only retaining rows were target matches RNAi
KD.tbl <-
  KD.tbl[which(mapply(grepl, KD.tbl$tx_name, KD.tbl$target)), ]

# Concatenating ID and target for plotting
KD.tbl$tx_name_target <-
  paste(KD.tbl$tx_name, KD.tbl$target, sep = "_")

# Summarizing by group and calculating SEM
KD.tbl <- group_by(KD.tbl, tx_name_target) %>%
    dplyr::summarise(mean_FC = mean(FC),
            sd_FC = sd(FC))

# Factoring levels for plotting
KD.tbl$tx_name_target <-
    factor(
        KD.tbl$tx_name_target,
        levels = c(
            "GAF_GAF",
            "BAP_BAP",
            "NURF_NURF",
            "BAP_NURFBAP",
            "NURF_NURFBAP"
        )
    )

# Plotting barplot
S1A <- ggplot(KD.tbl,
              aes(x = tx_name_target, y = mean_FC, fill = tx_name_target)
              )+
    geom_col(show.legend = F)+
    geom_errorbar(
      aes(ymin = mean_FC - sd_FC, ymax = mean_FC + sd_FC), width = 0.1
      )+
    ggtheme.jj() +
    scale_fill_manual(values = c(
        "#66CCEE",
        "#228833", # Manual color scaling because
        "#CCBB44", # the NURF+BAP condition has
        "#AA3377", # two values in this ploy and must
        "#AA3377"  # be entered twice
    )) +
    scale_y_continuous(limits = c(0, 1.0), expand = c(0, 0))+
    scale_x_discrete(
      labels = c("GAF", "BAP170", "NURF301", "N+B-BAP", "N+B-NURF")
      ) +
    ylab("Fraction mRNA") +
    xlab("RNAi")+
    theme(
        axis.line.x = element_blank(),
        axis.ticks.x = element_blank(),
        axis.text.x = element_text(angle = 45, hjust = 1)
    )
```

```
# Filtering out genes from the RNA-seq counts table that
# aren't in the PRO-seq table (blacklisted due to too much
# US transcription). This has to wait until now because
# Trl-RA is one of the genes that gets filtered
RNA.tbl <-
  RNA.tbl[which(!(RNA.tbl$tx_name %in% blacklist.chr)),]
RNA_normed.tbl <-
  RNA_normed.tbl[which(!(RNA_normed.tbl$tx_name %in% blacklist.chr)),]
```

Figure S2

A - PCA PRO-seq GB
B - PCA RNA-seq genes
C - PCA ATAC-seq peaks
D - PCA ATAC-seq promoters

```
# Panel A - PCA of PRO-seq signal in gene body regions
# Formatting count matrix for DESeq2. Rownames = gene names
GB.df <- data.frame(GB.tbl, row.names = "tx_name")

# colData object for DESeq2
col_data.df <- data.frame(
  row.names = colnames(GB.df),
  RNAi = c(
    "BAP170",
    "BAP170",
    "GAF",
    "GAF",
    "LACZ",
    "LACZ",
    "NURF301BAP170",
    "NURF301BAP170",
    "NURF301",
    "NURF301"
  )
)

# Construction of DESeq2 object
GB.dds <- DESeqDataSetFromMatrix(countData = GB.df,
                                 colData = col_data.df,
                                 design = ~ RNAi)

# Adding spike-in scale factors
sizeFactors(GB.dds) <- scale_facts_PROseq.dbl

# Running DESeq2
GB.dds <- DESeq(GB.dds)

# Variance stabilizing log transform
GB.rld <- rlog(GB.dds, blind = F)

# Getting PCA data
```

```
GB.pca <-
  plotPCA(GB.rld,
          intgroup = c("RNAi"),
          returnData = TRUE)

# Getting % Var explained by each PC
GB_pcavar.dbl <- round(100 * attr(GB.pca, "percentVar"))

# Factoring for plotting
GB.pca$RNAi <-
  factor(GB.pca$RNAi,
         levels =
           c("LACZ", "GAF", "BAP170", "NURF301", "NURF301BAP170")
         )

# Plotting
S2A <- ggplot(GB.pca, aes(PC1, PC2, color = RNAi)) +
  geom_point(size = 2, show.legend = T) +
  xlab(paste0("PC1: ", GB_pcavar.dbl[1], "% var.")) +
  ylab(paste0("PC2: ", GB_pcavar.dbl[2], "% var.")) +
  ggtheme.jj() +
  scale_color_manual(
    values = RNAi_cols.chr,
    breaks = c("LACZ", "GAF", "BAP170", "NURF301", "NURF301BAP170"),
    labels = c("LACZ", "GAF", "BAP", "NURF", "N+B")
  ) +
  scale_x_continuous(
    limits = c(-20, 30),
    expand = c(0, 0),
    breaks = c(-20, -10, 0, 10, 20, 30)
  ) +
  scale_y_continuous(
    limits = c(-20, 30),
    expand = c(0, 0),
    breaks = c(-20,-10, 0, 10, 20, 30)
  ) +
  ggtitle("Gene Body PRO-seq")

# Panel B - PCA of RNA-seq counts per gene
# Formatting count matrix for DESeq2. Rownames = gene names
RNA.tbl <- RNA.tbl[which(!(RNA.tbl$tx_name %in% blacklist.chr)), ]
RNA.df <- data.frame(RNA.tbl, row.names = "tx_name")

# colData object for DESeq2
col_data.df <- data.frame(
  row.names = colnames(RNA.df),
  RNAi = c(
    "BAP170",
    "BAP170",
    "GAF",
    "GAF",
    "LACZ",
    "LACZ",
```

```
    "NURF301BAP170",
    "NURF301BAP170",
    "NURF301",
    "NURF301"
  )
)

# construction of DESeq2 object
RNA.dds <- DESeqDataSetFromMatrix(countData = RNA.df,
                                  colData = col_data.df,
                                  design = ~ RNAi)

# Adding spike-in scale factors
sizeFactors(RNA.dds) <- scale_facts_RNAseq.dbl

# Running DESeq2
RNA.dds <- DESeq(RNA.dds)

# Variance stabilizing log transform
RNA.rld <- rlog(RNA.dds, blind = F)

# Getting PCA data
RNA.pca <-
  plotPCA(RNA.rld,
          intgroup = c("RNAi"),
          returnData = TRUE)
# Getting % Var explained by each PC
RNA_pcavar.dbl <- round(100 * attr(RNA.pca, "percentVar"))

# Factoring for plotting
RNA.pca$RNAi <-
  factor(RNA.pca$RNAi,
         levels =
           c("LACZ", "GAF", "BAP170", "NURF301", "NURF301BAP170")
         )

# Plotting
S2B <- ggplot(RNA.pca, aes(PC1, PC2, color = RNAi)) +
  geom_point(size = 2) +
  xlab(paste0("PC1: ", RNA_pcavar.dbl[1], "% var.")) +
  ylab(paste0("PC2: ", RNA_pcavar.dbl[2], "% var.")) +
  ggtheme.jj() +
  scale_color_manual(
    values = RNAi_cols.chr,
    breaks = c("LACZ", "GAF", "BAP170", "NURF301", "NURF301BAP170"),
    labels = c("LACZ", "GAF", "BAP", "NURF", "N+B")
  ) +
  scale_x_continuous(
    limits = c(-20, 20),
    expand = c(0, 0),
    breaks = c(-20, -10, 0, 10, 20)
  ) +
  scale_y_continuous(
```

```
    limits = c(-15, 15),
    expand = c(0, 0),
    breaks = c(-15, 0, 15)
  ) +
  ggtitle("3'RNA-seq")

# Panel C - PCA of ATAC-seq counts in peaks called using macs2
# Formatting count matrix for DESeq2. Rownames = gene names
ATAC_peaks.df <- data.frame(ATAC_peaks.tbl, row.names = "peak_name")

# colData object for DESeq2
col_data.df <- data.frame(
  row.names = colnames(ATAC_peaks.df),
  RNAi = c(
    "BAP170",
    "BAP170",
    "GAF",
    "GAF",
    "LACZ",
    "LACZ",
    "NURF301BAP170",
    "NURF301BAP170",
    "NURF301",
    "NURF301"
  )
)

# Construction of DESeq2 object
ATAC_peaks.dds <- DESeqDataSetFromMatrix(countData = ATAC_peaks.df,
                                         colData = col_data.df,
                                         design = ~ RNAi)

# Using DESeq2 internal normalization for ATAC-seq data

# Running DESeq2
ATAC_peaks.dds <- DESeq(ATAC_peaks.dds)

# Variance stabilizing log transform
ATAC_peaks.rld <- rlog(ATAC_peaks.dds, blind = F)

# Getting PCA data
ATAC_peaks.pca <-
  plotPCA(ATAC_peaks.rld,
          intgroup = c("RNAi"),
          returnData = TRUE)
# Getting % Var explained by each PC
ATAC_peaks_pcavar.dbl <-
  round(100 * attr(ATAC_peaks.pca, "percentVar"))

# Factoring for plotting
ATAC_peaks.pca$RNAi <-
  factor(ATAC_peaks.pca$RNAi,
         levels =
```

```
          c("LACZ", "GAF", "BAP170", "NURF301", "NURF301BAP170")
        )

# Plotting
S2C <- ggplot(ATAC_peaks.pca, aes(PC1, PC2, color = RNAi)) +
  geom_point(size = 2, show.legend = T) +
  xlab(paste0("PC1: ", ATAC_peaks_pcavar.dbl[1], "% var.")) +
  ylab(paste0("PC2: ", ATAC_peaks_pcavar.dbl[2], "% var.")) +
  ggtheme.jj() +
  scale_color_manual(
    values = RNAi_cols.chr,
    breaks = c("LACZ", "GAF", "BAP170", "NURF301", "NURF301BAP170"),
    labels = c("LACZ", "GAF", "BAP", "NURF", "N+B")
  )+
scale_x_continuous(
  limits = c(-40, 40),
  expand = c(0, 0),
  breaks = c(-40,-20, 0, 20, 40)
) +
  scale_y_continuous(
    limits = c(-20, 20),
    expand = c(0, 0),
    breaks = c(-20,-10, 0, 10, 20)
  ) +
  ggtitle("ATAC-seq (peaks)")


# Panel D - PCA of ATAC-seq counts in promoter regions
# Getting counts matrix
ATAC_PR.df <- data.frame(ATAC_PR.tbl, row.names = "tx_name")

# colData object for DESeq2
col_data.df <- data.frame(
  row.names = colnames(ATAC_PR.df),
  RNAi = c(
    "BAP170",
    "BAP170",
    "GAF",
    "GAF",
    "LACZ",
    "LACZ",
    "NURF301BAP170",
    "NURF301BAP170",
    "NURF301",
    "NURF301"
  )
)

# construction of DESeq2 object
ATAC_PR.dds <- DESeqDataSetFromMatrix(countData = ATAC_PR.df,
                                      colData = col_data.df,
                                      design = ~ RNAi)
```

```
# Using DESeq2 internal normalization for ATAC-seq data

# Running DESeq2
ATAC_PR.dds <- DESeq(ATAC_PR.dds)

# Variance stabilizing log transform
ATAC_PR.rld <- rlog(ATAC_PR.dds, blind = F)

# Getting PCA data
ATAC_PR.pca <-
  plotPCA(ATAC_PR.rld,
          intgroup = c("RNAi"),
          returnData = TRUE)
# Getting % Var explained by each PC
ATAC_PR_pcavar.dbl <- round(100 * attr(ATAC_PR.pca, "percentVar"))

# Factoring for plotting
ATAC_PR.pca$RNAi <-
  factor(ATAC_PR.pca$RNAi,
         levels = c("LACZ", "GAF", "BAP170", "NURF301", "NURF301BAP170"))

# Plotting
S2D <- ggplot(ATAC_PR.pca, aes(PC1, PC2, color = RNAi)) +
    geom_point(size = 2) +
    xlab(paste0("PC1: ", ATAC_PR_pcavar.dbl[1], "% var.")) +
    ylab(paste0("PC2: ", ATAC_PR_pcavar.dbl[2], "% var.")) +
    ggtheme.jj() +
    scale_color_manual(values = RNAi_cols.chr,
                       breaks = c("LACZ", "GAF", "BAP170", "NURF301", "NURF301BAP170"),
                       labels = c("LACZ", "GAF", "BAP", "NURF", "N+B"))+
    scale_x_continuous(limits = c(-25, 25),
                       expand = c(0,0),
                       breaks = c(-25, -12.5, 0, 12.5, 25))+
        scale_y_continuous(limits = c(-15, 15),
                       expand = c(0,0),
                       breaks = c(-15, -7.5, 0, 7.5, 15))+
      ggtitle("ATAC-seq (promoters)")

# Combining panels
S2_all <- (S2A + S2B) / (S2C + S2D) +
    plot_layout(guides = 'collect')+
    plot_annotation(tag_levels = 'A') &
    theme(plot.tag = element_text(family = "Helvetica", face = "bold", size = 12))

# Saving ATACseq scale factors for later
scale_facts_ATACDHS.dbl <- sizeFactors(ATAC_PR.dds)
scale_facts_ATACDHS.lst <- setNames(as.list(scale_facts_ATACDHS.dbl), samples_ATACDHS.chr)
```

Figure S3

A - scatter plots PRO-seq GB raw
B - scatter plots PRO-seq GB normalized

C - scatter plots PRO-seq PR raw
D - scatter plots PRO-seq PR normalized

```
# Panel A - scatter plots PRO-seq GB raw
S3A <- rep_corr_scatter.jj(
  df = GB.tbl,
  sep_colnames_into = c("condition", NA, "rep")) +
  scale_y_log10(limits = c(1, 1000000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000, 1000000),
                labels = c(0, 1, 2, 3, 4, 5, 6))+
  scale_x_log10(limits = c(1, 1000000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000, 1000000),
                labels = c(0, 1, 2, 3, 4, 5, 6))+
  facet_grid(.~ condition)+
  xlab("log10 Rep. 1") +
  ylab("log10 Rep. 2") +
  ggtitle("PRO-seq Gene Body Raw Counts")+
  guides(fill = guide_colourbar(barwidth = 0.5, barheight = 5))

# Panel B - scatter plots PRO-seq GB normalized
S3B <- rep_corr_scatter.jj(
  df = GB_normed.tbl,
  sep_colnames_into = c("condition", NA, "rep")) +
  scale_y_log10(limits = c(1, 1000000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000, 1000000),
                labels = c(0, 1, 2, 3, 4, 5, 6))+
  scale_x_log10(limits = c(1, 1000000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000, 1000000),
                labels = c(0, 1, 2, 3, 4, 5, 6))+
  facet_grid(.~ condition)+
  xlab("log10 Rep. 1") +
  ylab("log10 Rep. 2") +
  ggtitle("PRO-seq Gene Body Spike-In Normalized Counts")+
  guides(fill = guide_colourbar(barwidth = 0.5, barheight = 5))

# Panel C - scatter plots PRO-seq PR raw
S3C <- rep_corr_scatter.jj(
  df = PR.tbl,
  sep_colnames_into = c("condition", NA, "rep")) +
  scale_y_log10(limits = c(1, 100000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000),
                labels = c(0, 1, 2, 3, 4, 5))+
  scale_x_log10(limits = c(1, 100000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000),
                labels = c(0, 1, 2, 3, 4, 5))+
  facet_grid(.~ condition)+
  xlab("log10 Rep. 1") +
  ylab("log10 Rep. 2") +
  ggtitle("PRO-seq Promoter Raw Counts")+
  guides(fill = guide_colourbar(barwidth = 0.5, barheight = 5))

# Panel D - scatter plots PRO-seq PR normalized
S3D <- rep_corr_scatter.jj(
  df = PR_normed.tbl,
```

```
  sep_colnames_into = c("condition", NA, "rep")) +
  scale_y_log10(limits = c(1, 100000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000),
                labels = c(0, 1, 2, 3, 4, 5))+
  scale_x_log10(limits = c(1, 100000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000),
                labels = c(0, 1, 2, 3, 4, 5))+
  facet_grid(.~ condition)+
  xlab("log10 Rep. 1") +
  ylab("log10 Rep. 2") +
  ggtitle("PRO-seq Promoter Spike-In Normalized Counts")+
  guides(fill = guide_colourbar(barwidth = 0.5, barheight = 5))

# Combining panels
S3_all <- S3A / S3B / S3C / S3D  +
    plot_annotation(tag_levels = 'A') &
    theme(plot.tag = element_text(family = "Helvetica",
                                  face = "bold",
                                  size = 12))
```

Figure S4

A - scatter plots RNA-seq raw
B - scatter plots RNA-seq normalized
C - scatter plots ATAC-seq raw
D - scatter plots ATAC-seq normalized

```
# Panel A - scatter plots RNA-seq raw
S4A <- rep_corr_scatter.jj(
  df = RNA.tbl,
  sep_colnames_into = c("condition", NA, "rep")) +
  scale_y_log10(limits = c(1, 100000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000),
                labels = c(0, 1, 2, 3, 4, 5))+
  scale_x_log10(limits = c(1, 100000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000),
                labels = c(0, 1, 2, 3, 4, 5))+
  facet_grid(.~ condition)+
  xlab("log10 Rep. 1") +
  ylab("log10 Rep. 2") +
  ggtitle("3'RNA-seq Raw Counts")+
  guides(fill = guide_colourbar(barwidth = 0.5, barheight = 5))

# Panel B - scatter plots RNA-seq normalized
S4B <- rep_corr_scatter.jj(
  df = RNA_normed.tbl,
  sep_colnames_into = c("condition", NA, "rep")) +
  scale_y_log10(limits = c(1, 100000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000),
                labels = c(0, 1, 2, 3, 4, 5))+
  scale_x_log10(limits = c(1, 100000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000),
                labels = c(0, 1, 2, 3, 4, 5))+
```

```
  facet_grid(.~ condition)+
  xlab("log10 Rep. 1") +
  ylab("log10 Rep. 2") +
  ggtitle("3'RNA-seq Spike-In Normalized Counts")+
  guides(fill = guide_colourbar(barwidth = 0.5, barheight = 5))

# Panel C - scatter plots ATAC-seq raw
S4C <- rep_corr_scatter.jj(
  df = ATAC_PR.tbl,
  sep_colnames_into = c("condition", NA, "rep")) +
  scale_y_log10(limits = c(1, 1000000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000, 1000000),
                labels = c(0, 1, 2, 3, 4, 5, 6))+
  scale_x_log10(limits = c(1, 1000000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000, 1000000),
                labels = c(0, 1, 2, 3, 4, 5, 6))+
  facet_grid(.~ condition)+
  xlab("log10 Rep. 1") +
  ylab("log10 Rep. 2") +
  ggtitle("ATAC-seq Promoter Raw Counts")+
  guides(fill = guide_colourbar(barwidth = 0.5, barheight = 5))

# Panel D - scatter plots ATAC-seq normalized

## Normalizing ATAC-seq data
ATAC_PR_normed.tbl <- ATAC_PR.tbl
for (i in colnames(ATAC_PR.tbl[, -1])) {
    ATAC_PR_normed.tbl[, i] <- ATAC_PR.tbl[, i] / scale_facts_ATACDHS.lst[[i]]
}

# Plotting
S4D <- rep_corr_scatter.jj(
  df = ATAC_PR_normed.tbl,
  sep_colnames_into = c("condition", NA, "rep")) +
  scale_y_log10(limits = c(1, 1000000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000, 1000000),
                labels = c(0, 1, 2, 3, 4, 5, 6))+
  scale_x_log10(limits = c(1, 1000000), expand = c(0,0),
                breaks = c(1, 10, 100, 1000, 10000, 100000, 1000000),
                labels = c(0, 1, 2, 3, 4, 5, 6))+
  facet_grid(.~ condition)+
  xlab("log10 Rep. 1") +
  ylab("log10 Rep. 2") +
  ggtitle("ATAC-seq Promoter Normalized Counts")+
  guides(fill = guide_colourbar(barwidth = 0.5, barheight = 5))

# Combining panels
S4_all <- S4A / S4B / S4C / S4D  + plot_annotation(tag_levels = 'A') &
  theme(plot.tag = element_text(family = "Helvetica",
                                face = "bold", size = 12))
```

Figure 2

A - DREME logos from GAF bound unchanged PR vs GAF bound changed PR
B - ChIP-seq, PRO-seq and ATAC-seq at classified GAF promoters

```
# A - DREME logos from GAF bound unchanged PR vs GAF bound changed PR
# Defining set of GAF bound genes, classified by whether they
# have decreased pausing upon GAF knockdown

# Getting list of genes and distance of each promoter
# to nearest GAF peak
gafBoundGenes.gr <- genes.gr
gafBoundGenes.gr$distToGAFpeak <-
  mcols(distanceToNearest(
    promoters(gafBoundGenes.gr, 500, 0),
    GAF_peaks.gr))$distance

# subsetting to genes where dist == 0, so GAF peak within TSS-500 to TSS)
gafBoundGenes.gr <-
  gafBoundGenes.gr[gafBoundGenes.gr$distToGAFpeak == 0, ]

# Dividing based on behavior of promoter upon GAF-RNAi
gafBoundGenes_gafDepPR.gr <-
  gafBoundGenes.gr[gafBoundGenes.gr$tx_name %in% gafDepPR.gr$tx_name, ]
gafBoundGenes_gafIndPR.gr <-
  gafBoundGenes.gr[!gafBoundGenes.gr$tx_name %in% gafDepPR.gr$tx_name, ]

# Use these as input for DREME (external) and insert motifs into figure
write_tsv(
  as.data.frame(
    promoters(
      gafBoundGenes_gafDepPR.gr, 500, 0)),
    "bed/gafBoundGenes_gafDepPR.bed")
write_tsv(
  as.data.frame(
    promoters(
      gafBoundGenes_gafIndPR.gr, 500, 0)),
    "bed/gafBoundGenes_gafIndPR.bed")

# Getting GAF ChIP signal intensity for
# gafBoundGenes_gafDepPR.gr (for sorting later)
gafBoundGenes_gafDepPR.gr$GAF_signal <-
  getCountsByRegions(
    ChIP.lst$GAF_ChIPseq,
    promoters(gafBoundGenes_gafDepPR.gr, 500, 0),
    expand_ranges = TRUE
  )

# B - ChIP-seq, PRO-seq and ATAC-seq at classified GAF promoters
# Adding total ChIP-seq signal for M1BP and BEAF32 to genes
gafBoundGenes_gafIndPR.gr$M1BP_signal <-
  getCountsByRegions(
    ChIP.lst$M1BP_ChIPseq,
    promoters(gafBoundGenes_gafIndPR.gr, 500, 0),
```

```
    expand_ranges = TRUE
  )

gafBoundGenes_gafIndPR.gr$BEAF32_signal <-
  getCountsByRegions(
    ChIP.lst$BEAF32_ChIPseq,
    promoters(gafBoundGenes_gafIndPR.gr, 500, 0),
    expand_ranges = TRUE
  )

# Defining classes, considering a promoter "bound" by each factor
# if it's in the top 25% of promoters for signal of that factor by ChIP-seq
gafBoundGenes_gafIndPR.gr$class <- ifelse(
  gafBoundGenes_gafIndPR.gr$M1BP_signal > quantile(
    gafBoundGenes_gafIndPR.gr$M1BP_signal)[[4]],
  ifelse(
    gafBoundGenes_gafIndPR.gr$BEAF32_signal > quantile(
      gafBoundGenes_gafIndPR.gr$BEAF32_signal)[[4]],
    "Both",
    "M1BP"
  ),
  ifelse(
    gafBoundGenes_gafIndPR.gr$BEAF32_signal > quantile(
      gafBoundGenes_gafIndPR.gr$BEAF32_signal)[[4]],
    "BEAF32",
    NA
))

# Dropping promoters that aren't either M1BP or BEAF32 bound
gafBoundGenes_gafIndPR.gr <-
  gafBoundGenes_gafIndPR.gr[!is.na(gafBoundGenes_gafIndPR.gr$class),]

# Concatenating and sorting GRs:
# 1 - GAFdepGAFbound sorted by GAF intensity
# 2 - GAFindGAFbound M1BP/BEAF32 bound sorted by M1BP intensity
# 3 - GAFindGAFbound M1BP bound sorted by M1BP intensity
# 4 - GAFindGAFbound BEAF32 bound sorted by BEAF32 intensity
gafDepPR_sorted.gr <-
  c(
    sort(gafBoundGenes_gafDepPR.gr,
         by = ~ GAF_signal, decreasing = TRUE),
    sort(
      gafBoundGenes_gafIndPR.gr[
        gafBoundGenes_gafIndPR.gr$class == "Both",
        ],
      by = ~ M1BP_signal, decreasing = TRUE),
    sort(
      gafBoundGenes_gafIndPR.gr[
        gafBoundGenes_gafIndPR.gr$class == "M1BP",
        ],
      by = ~ M1BP_signal, decreasing = TRUE),
    sort(
      gafBoundGenes_gafIndPR.gr[
```

```
        gafBoundGenes_gafIndPR.gr$class == "BEAF32",
        ],
      by = ~ BEAF32_signal, decreasing = TRUE)
  )

# Getting ChIP signal for hm
gafBoundGenesSorted_ChIP.mat <- cbind(
  getCountsByPositions(
    dataset.gr = ChIP.lst$GAF_ChIPseq,
    regions.gr = promoters(gafDepPR_sorted.gr, 500, 500),
    binsize = 10,
    expand_ranges = TRUE
  ),
  getCountsByPositions(
    dataset.gr = ChIP.lst$M1BP_ChIPseq,
    regions.gr = promoters(gafDepPR_sorted.gr, 500, 500),
    binsize = 10,
    expand_ranges = TRUE
  ),
  getCountsByPositions(
    dataset.gr = ChIP.lst$BEAF32_ChIPseq,
    regions.gr = promoters(gafDepPR_sorted.gr, 500, 500),
    binsize = 10,
    expand_ranges = TRUE
  )
)

# Setting up vectors to use for splitting heatmap rows and cols
row_split <-
  c(
    rep(
      "GAF",
      length(gafBoundGenes_gafDepPR.gr)
      ),
    rep(
      "Both",
      length(gafBoundGenes_gafIndPR.gr[
        gafBoundGenes_gafIndPR.gr$class == "Both",
                                       ])
      ),
    rep(
      "M1BP",
      length(gafBoundGenes_gafIndPR.gr[
        gafBoundGenes_gafIndPR.gr$class == "M1BP",
                                       ])
      ),
    rep(
      "BEAF32",
      length(gafBoundGenes_gafIndPR.gr[
        gafBoundGenes_gafIndPR.gr$class == "BEAF32",
                                       ])
      )
  )
```

```
row_split <- factor(row_split,
                    levels = c("GAF", "Both", "M1BP", "BEAF32"))
col_fun = colorRamp2(c(0, 50, 100), viridis(3))

# Plotting Heatmap
F2Ba <- Heatmap(
  gafBoundGenesSorted_ChIP.mat,
  cluster_columns = F,
  cluster_rows = F,
  col = col_fun,
  row_split = row_split
)

# Adding PRO-seq and ATAC-seq data to GR
gafDepPR_sorted.gr$LACZ_PROseq <-
  getCountsByRegions(
    PROseq_normed_merged.lst$LACZ_PROseq,
    promoters(gafDepPR_sorted.gr,50,100),
    expand_ranges = FALSE
  )

gafDepPR_sorted.gr$GAF_PROseq <-
  getCountsByRegions(
    PROseq_normed_merged.lst$GAF_PROseq,
    promoters(gafDepPR_sorted.gr,50,100),
    expand_ranges = FALSE
  )

gafDepPR_sorted.gr$BAP170_PROseq <-
  getCountsByRegions(
    PROseq_normed_merged.lst$BAP170_PROseq,
    promoters(gafDepPR_sorted.gr,50,100),
    expand_ranges = FALSE
  )

gafDepPR_sorted.gr$NURF301_PROseq <-
  getCountsByRegions(
    PROseq_normed_merged.lst$NURF301_PROseq,
    promoters(gafDepPR_sorted.gr,50,100),
    expand_ranges = FALSE
  )

gafDepPR_sorted.gr$NURF301BAP170_PROseq <-
  getCountsByRegions(
    PROseq_normed_merged.lst$NURF301BAP170_PROseq,
    promoters(gafDepPR_sorted.gr,50,100),
    expand_ranges = FALSE
  )

gafDepPR_sorted.gr$M1BPLACZ_PROseq <-
  getCountsByRegions(
    PROseq_M1BP.lst$LACZ,
    promoters(gafDepPR_sorted.gr,50,100),
```

```
    expand_ranges = FALSE
  )

gafDepPR_sorted.gr$M1BP_PROseq <-
  getCountsByRegions(
    PROseq_M1BP.lst$M1BP,
    promoters(gafDepPR_sorted.gr,50,100),
    expand_ranges = FALSE
  )

# Getting matrix of l2FC of each RNAi vs LACZ
# (adding pseudocount to prevent Inf/NA values)
gafDepPR_sorted_PROseql2FC.mat <- cbind(
  log2(
    (gafDepPR_sorted.gr$GAF_PROseq + 1) /
      (gafDepPR_sorted.gr$LACZ_PROseq + 1)
    ),
  log2(
    (gafDepPR_sorted.gr$BAP170_PROseq + 1) /
      (gafDepPR_sorted.gr$LACZ_PROseq + 1)
    ),
  log2(
    (gafDepPR_sorted.gr$NURF301_PROseq + 1) /
      (gafDepPR_sorted.gr$LACZ_PROseq + 1)
    ),
  log2(
    (gafDepPR_sorted.gr$NURF301BAP170_PROseq + 1) /
      (gafDepPR_sorted.gr$LACZ_PROseq + 1)
    )
)

# Plotting Heatmap
col_fun_fc = colorRamp2(c(-2, 0, 2), c("#3B4992", "ghostwhite", "#BB0021"))
F2Bb <- Heatmap(
  gafDepPR_sorted_PROseql2FC.mat,
  cluster_columns = F,
  cluster_rows = F,
  col = col_fun_fc,
  row_split = row_split
)

# Plotting heatmap of M1BP l2fC pro-seq promoter
gafDepPR_sorted_M1BPPROseql2FC.mat <-
  log2((
    gafDepPR_sorted.gr$M1BP_PROseq + 1) /
      (gafDepPR_sorted.gr$M1BPLACZ_PROseq+ 1)
    )


F2Bc <- Heatmap(
  gafDepPR_sorted_M1BPPROseql2FC.mat,
  cluster_columns = F,
  cluster_rows = F,
```

```
  col = col_fun_fc,
  row_split = row_split,
  column_labels = c("")
)

# Adding ATAC-seq data
gafDepPR_sorted.gr$LACZ_ATACseq <-
  getCountsByRegions(
    ATACDHS_normed_merged.lst$LACZ_ATACDHS,
    promoters(gafDepPR_sorted.gr, 250, 0),
    expand_ranges = TRUE
  )

gafDepPR_sorted.gr$GAF_ATACseq <-
  getCountsByRegions(
    ATACDHS_normed_merged.lst$GAF_ATACDHS,
    promoters(gafDepPR_sorted.gr,250, 0),
    expand_ranges = TRUE
  )

gafDepPR_sorted.gr$BAP170_ATACseq <-
  getCountsByRegions(
    ATACDHS_normed_merged.lst$BAP170_ATACDHS,
    promoters(gafDepPR_sorted.gr,250, 0),
    expand_ranges = TRUE
  )

gafDepPR_sorted.gr$NURF301_ATACseq <-
  getCountsByRegions(
    ATACDHS_normed_merged.lst$NURF301_ATACDHS,
    promoters(gafDepPR_sorted.gr,250, 0),
    expand_ranges = TRUE
  )

gafDepPR_sorted.gr$NURF301BAP170_ATACseq <-
  getCountsByRegions(
    ATACDHS_normed_merged.lst$NURF301BAP170_ATACDHS,
    promoters(gafDepPR_sorted.gr,250, 0),
    expand_ranges = TRUE
  )

# Getting matrix of l2FC of each RNAi vs LACZ
gafDepPR_sorted_ATACseql2FC.mat <- cbind(
  log2(
    (gafDepPR_sorted.gr$GAF_ATACseq + 1) /
      (gafDepPR_sorted.gr$LACZ_ATACseq + 1)
    ),
  log2(
    (gafDepPR_sorted.gr$BAP170_ATACseq + 1) /
      (gafDepPR_sorted.gr$LACZ_ATACseq + 1)
    ),
  log2(
    (gafDepPR_sorted.gr$NURF301_ATACseq + 1) /
```

```
      (gafDepPR_sorted.gr$LACZ_ATACseq + 1)
    ),
  log2(
    (gafDepPR_sorted.gr$NURF301BAP170_ATACseq + 1) /
      (gafDepPR_sorted.gr$LACZ_ATACseq + 1)
    )
)

# Plotting Heatmap
F2Bd <- Heatmap(
  gafDepPR_sorted_ATACseql2FC.mat,
  cluster_columns = F,
  cluster_rows = F,
  col = col_fun_fc,
  row_split = row_split
)
```

Figure S5

A-D - MA plots promoter PRO-seq
E-H - MA plots GB PRO-seq
I-L - MA plots of RNA-seq

```
# A-D - MA plots promoter PRO-seq
# Running DESeq2 for remaining Promoter comparisons
PR_NURFBAP.res <-
  subset_DESeq.jj(
    PR.df, PR_col_data.df,
    scale_facts_PROseq.dbl,
    c(5,6,7,8),
    c("RNAi", "NURFBAP", "LACZ"),
    ~RNAi)

# Plotting
S5A <- maplot.jj(PR_GAF.res)
S5B <- maplot.jj(PR_BAP.res)
S5C <- maplot.jj(PR_NURF.res)
S5D <- maplot.jj(PR_NURFBAP.res)


# E-H - MA plots GB PRO-seq
# Running DESeq2 for all GB regions
GB.df <- data.frame(GB.tbl, row.names = "tx_name")
GB_col_data.df <- data.frame(
  row.names = colnames(GB.df),
  RNAi = c(
    "BAP170",
    "BAP170",
    "GAF",
    "GAF",
    "LACZ",
    "LACZ",
    "NURFBAP",
```

```
    "NURFBAP",
    "NURF301",
    "NURF301"
  )
)
GB_GAF.res <-
  subset_DESeq.jj(
    GB.df,
    GB_col_data.df,
    scale_facts_PROseq.dbl,
    c(3,4,5,6),
    c("RNAi", "GAF", "LACZ"),
    ~RNAi)
GB_BAP.res <-
  subset_DESeq.jj(
    GB.df,
    GB_col_data.df,
    scale_facts_PROseq.dbl,
    c(1,2,5,6),
    c("RNAi", "BAP170", "LACZ"),
    ~RNAi)
GB_NURF.res <-
  subset_DESeq.jj(
    GB.df,
    GB_col_data.df,
    scale_facts_PROseq.dbl,
    c(5,6,9,10),
    c("RNAi", "NURF301", "LACZ"),
    ~RNAi)
GB_NURFBAP.res <-
  subset_DESeq.jj(
    GB.df,
    GB_col_data.df,
    scale_facts_PROseq.dbl,
    c(5,6,7,8),
    c("RNAi", "NURFBAP", "LACZ"),
    ~RNAi)

S5E <- maplot.jj(GB_GAF.res)
S5F <- maplot.jj(GB_BAP.res)
S5G <- maplot.jj(GB_NURF.res)
S5H <- maplot.jj(GB_NURFBAP.res)

# I-L - MA plots of RNA-seq
# Running DESeq2 for all RNA-seq
RNA.df <- data.frame(RNA.tbl, row.names = "tx_name")
RNA_col_data.df <- data.frame(
  row.names = colnames(RNA.df),
  RNAi = c(
    "BAP170",
    "BAP170",
    "GAF",
    "GAF",
```

```
    "LACZ",
    "LACZ",
    "NURFBAP",
    "NURFBAP",
    "NURF301",
    "NURF301"
  )
)
RNA_GAF.res <-
  subset_DESeq.jj(
    RNA.df,
    RNA_col_data.df,
    scale_facts_RNAseq.dbl,
    c(3,4,5,6),
    c("RNAi", "GAF", "LACZ"),
    ~RNAi)
RNA_BAP.res <-
  subset_DESeq.jj(
    RNA.df,
    RNA_col_data.df,
    scale_facts_RNAseq.dbl,
    c(1,2,5,6),
    c("RNAi", "BAP170", "LACZ"),
    ~RNAi)
RNA_NURF.res <-
  subset_DESeq.jj(
    RNA.df,
    RNA_col_data.df,
    scale_facts_RNAseq.dbl,
    c(5,6,9,10),
    c("RNAi", "NURF301", "LACZ"),
    ~RNAi)
RNA_NURFBAP.res <-
  subset_DESeq.jj(
    RNA.df,
    RNA_col_data.df,
    scale_facts_RNAseq.dbl,
    c(5,6,7,8),
    c("RNAi", "NURFBAP", "LACZ"),
    ~RNAi)

S5I <- maplot.jj(RNA_GAF.res)
S5J <- maplot.jj(RNA_BAP.res)
S5K <- maplot.jj(RNA_NURF.res)
S5L <- maplot.jj(RNA_NURFBAP.res)

# Combining all panels
S5_all <- (S5A | S5B | S5C | S5D) /
  (S5E | S5F | S5G | S5H) /
  (S5I | S5J | S5K | S5L) +
  plot_annotation(tag_levels = 'A') &
    theme(plot.tag = element_text(
      family = "Helvetica",
```

```
      face = "bold",
      size = 12))

# Writing DESeq2 results to files
# Helper function
res_write_tsv.jj <-
  function(res, filename) {
    res.df <- as.data.frame(res)
    res.df$tx_name <- row.names(res.df)
    write_tsv(res.df, filename)
  }

res_write_tsv.jj(PR_GAF.res, "DESeq_results/GAF_PR_res.tsv")
res_write_tsv.jj(PR_BAP.res, "DESeq_results/BAP_PR_res.tsv")
res_write_tsv.jj(PR_NURF.res, "DESeq_results/NURF_PR_res.tsv")
res_write_tsv.jj(PR_NURFBAP.res, "DESeq_results/NURFBAP_PR_res.tsv")
res_write_tsv.jj(GB_GAF.res, "DESeq_results/GAF_GB_res.tsv")
res_write_tsv.jj(GB_BAP.res, "DESeq_results/BAP_GB_res.tsv")
res_write_tsv.jj(GB_NURF.res, "DESeq_results/NURF_GB_res.tsv")
res_write_tsv.jj(GB_NURFBAP.res, "DESeq_results/NURFBAP_GB_res.tsv")
res_write_tsv.jj(RNA_GAF.res, "DESeq_results/GAF_RNA_res.tsv")
res_write_tsv.jj(RNA_BAP.res, "DESeq_results/BAP_RNA_res.tsv")
res_write_tsv.jj(RNA_NURF.res, "DESeq_results/NURF_RNA_res.tsv")
res_write_tsv.jj(RNA_NURFBAP.res, "DESeq_results/NURFBAP_RNA_res.tsv")
```

Figure S6

A - Browser shot of E23-RC
B - Browser shot of Cyp9c1
C - Browser shot of Fatp3-RA
D - Browser shot of geko RB
E - Browser shot of out-RA
Browser shots all made with R package found here:
https://github.com/JAJ256/browser_plot.R
Analysis in this paper performed with commit: 1352d5c

```
# A - E23-RC
E23.gr <- genes.gr[which(genes.gr$tx_name == "E23-RC")]

# moving end of gene in (to condense plot)
start(E23.gr) <- start(E23.gr) + 10000

# Plotting (see file browserPlotR.R for more detail)
S6A <- browser_plotter.jj(
    E23.gr,
    list("PROseq" = PROseq_normed_merged.lst,
         "ATACDHS" = ATACDHS_normed_merged.lst,
         "ChIPseq" = ChIP.lst["GAF_ChIPseq"]),
    labs = c("LACZ", "GAF", "BAP", "NURF", "N+B",
             "LACZ", "GAF", "BAP", "NURF", "N+B",
             "GAF ChIP"),
    scale_bar_size = 1000,
```

```
    pad_left = 0,
    pad_right = 500,
    binsize = 1,
    bin_FUN = mean,
    .expand_ranges = list(
        "PROseq" = FALSE,
        "ATACDHS" = TRUE,
        "ChIPseq" = TRUE)
    )

# B - Cyp9c1
Cyp9c1.gr <- genes.gr[which(genes.gr$tx_name == "Cyp9c1-RA")]

# Plotting (see file browserPlotR.R for more detail)
S6B <- browser_plotter.jj(
    Cyp9c1.gr,
    list("PROseq" = PROseq_normed_merged.lst,
         "ATACDHS" = ATACDHS_normed_merged.lst,
         "ChIPseq" = ChIP.lst["GAF_ChIPseq"]),
    labs = c("LACZ", "GAF", "BAP", "NURF", "N+B",
             "LACZ", "GAF", "BAP", "NURF", "N+B",
             "GAF ChIP"),
    scale_bar_size = 1000,
    pad_left = 500,
    pad_right = 0,
    binsize = 1,
    bin_FUN = mean,
    .expand_ranges = list(
        "PROseq" = FALSE,
        "ATACDHS" = TRUE,
        "ChIPseq" = TRUE)
    )

# C - Fatp3-RA
Fatp3.gr <- genes.gr[which(genes.gr$tx_name == "Fatp3-RA")]

# Plotting (see file browserPlotR.R for more detail)
S6C <- browser_plotter.jj(
    Fatp3.gr,
    list("PROseq" = PROseq_normed_merged.lst,
         "ATACDHS" = ATACDHS_normed_merged.lst,
         "ChIPseq" = ChIP.lst["GAF_ChIPseq"]),
    labs = c("LACZ", "GAF", "BAP", "NURF", "N+B",
             "LACZ", "GAF", "BAP", "NURF", "N+B",
             "GAF ChIP"),
    scale_bar_size = 1000,
    pad_left = 500,
    pad_right = 0,
    binsize = 1,
    bin_FUN = mean,
    .expand_ranges = list(
        "PROseq" = FALSE,
        "ATACDHS" = TRUE,
```

```
        "ChIPseq" = TRUE)
    )

# D - geko RB
geko.gr <- genes.gr[which(genes.gr$tx_name == "geko-RB")]

# moving end of gene in (to condense plot)
end(geko.gr) <- end(geko.gr) - 3000

# Plotting (see file browserPlotR.R for more detail)
S6D <- browser_plotter.jj(
    geko.gr,
    list("PROseq" = PROseq_normed_merged.lst,
         "ATACDHS" = ATACDHS_normed_merged.lst,
         "ChIPseq" = ChIP.lst["GAF_ChIPseq"]),
    labs = c("LACZ", "GAF", "BAP", "NURF", "N+B",
             "LACZ", "GAF", "BAP", "NURF", "N+B",
             "GAF ChIP"),
    scale_bar_size = 1000,
    pad_left = 500,
    pad_right = 0,
    binsize = 1,
    bin_FUN = mean,
    .expand_ranges = list(
        "PROseq" = FALSE,
        "ATACDHS" = TRUE,
        "ChIPseq" = TRUE)
    )

# E - out-RA
out.gr <- genes.gr[which(genes.gr$tx_name == "out-RA")]

# moving end of gene in (to condense plot)
start(out.gr) <- start(out.gr) + 5000

# Plotting (see file browserPlotR.R for more detail)
S6E <- browser_plotter.jj(
    out.gr,
    list("PROseq" = PROseq_normed_merged.lst,
         "ATACDHS" = ATACDHS_normed_merged.lst,
         "ChIPseq" = ChIP.lst["GAF_ChIPseq"]),
    labs = c("LACZ", "GAF", "BAP", "NURF", "N+B",
             "LACZ", "GAF", "BAP", "NURF", "N+B",
             "GAF ChIP"),
    scale_bar_size = 1000,
    pad_left = 0,
    pad_right = 500,
    binsize = 1,
    bin_FUN = mean,
    .expand_ranges = list(
        "PROseq" = FALSE,
        "ATACDHS" = TRUE,
        "ChIPseq" = TRUE)
```

```
    )

S6_all <- grid.arrange(S6A, S6B, S6C, S6D, S6E, ncol = 3, nrow = 2)
```

Figure S7

A - ATAC-seq at BAP-dependent promoters
B - PRO-seq at BAP-dependent promoters (GAF/BAP/LACZ)
C - PRO-seq at BAP-dependent promoters (NURF/NURFBAP/LACZ)
D - GAF PR l2FC vs NURF+BAP l2FC

```
# A - ATAC-seq at BAP-dependent promoters
# Converting results object to df
PR_BAPres.df <- as.data.frame(PR_BAP.res)
PR_BAPres.df$tx_name <- row.names(PR_BAPres.df)

# Getting list of BAP-dependent promoters (FDR 0.01)
bapDepPR.gr <-
    genes.gr[which(
        genes.gr$tx_name %in%
            (PR_BAPres.df %>%
            filter(padj < 0.01 & log2FoldChange < 0))$tx_name
    )]

# Getting counts matrix of ATAC data
bapDepPR_ATACDHSSub.mat <-
    metaSubsample(
        dataset.gr = ATACDHS_normed_merged.lst,
        regions.gr = promoters(bapDepPR.gr, 500, 500),
        binsize = 1,
        first.output.xval = -500,
        expand_ranges = TRUE
    )

# Factoring sample.name column
bapDepPR_ATACDHSSub.mat$sample.name <-
    factor(
        bapDepPR_ATACDHSSub.mat$sample.name,
        levels = names(ATACDHS_normed_merged.lst)
    )

# Plotting
S7A <- ggMetaplot(bapDepPR_ATACDHSSub.mat) +
    gg_color_scale_RNAi_ATACDHS +
    gg_fill_scale_RNAi_ATACDHS +
    scale_x_continuous(
        limits = c(-500, 500),
        expand = c(0,0)
    ) +
    scale_y_continuous(
        limits = c(0, 200),
        expand = c(0,0)
```

```
    )+
  ggtitle("ATAC < 120 bp")

# B - PRO-seq at BAP-dependent promoters (GAF/BAP/LACZ)
# C - PRO-seq at BAP-dependent promoters (NURF/NURFBAP/LACZ)
# make one plot, duplicate in figure,
# and delete lines in illustrator

# Getting counts matrixs
bapDepPR_proSeqSub.mat <-
     metaSubsample(
        dataset.gr = PROseq_normed_merged.lst,
        regions.gr = promoters(bapDepPR.gr, 0, 150),
        binsize = 2,
        first.output.xval = 0,
        expand_ranges = FALSE
    )


bapDepGB_proSeqSub.mat <-
     metaSubsample(
        dataset.gr = PROseq_normed_merged.lst,
        regions.gr = promoters(bapDepPR.gr, 0, 1500),
        binsize = 20,
        first.output.xval = 0,
        expand_ranges = FALSE
    )

# Filling sample.name column and factoring
bapDepPR_proSeqSub.mat$sample.name <-
    factor(
        bapDepPR_proSeqSub.mat$sample.name,
        levels = names(PROseq_normed_merged.lst)
    )

bapDepGB_proSeqSub.mat$sample.name <-
    factor(
        bapDepGB_proSeqSub.mat$sample.name,
        levels = names(PROseq_normed_merged.lst)
    )


# Plotting promoter
S7Ba <- ggMetaplot(bapDepPR_proSeqSub.mat)+
    gg_color_scale_RNAi_PROseq +
    gg_fill_scale_RNAi_PROseq +
    scale_x_continuous(
        limits = c(0, 150),
        expand = c(0,0),
        breaks = c(0, 50, 100, 150)
    ) +
    scale_y_continuous(
        breaks = c(0, 50, 100)
```

```
    ) +
    coord_cartesian(
        ylim = c(0, 100),
        expand = 0,
        )+
  ggtitle("PRO-seq")


# Plotting gene body
S7Bb <- ggMetaplot(bapDepGB_proSeqSub.mat)+
    gg_color_scale_RNAi_PROseq +
    gg_fill_scale_RNAi_PROseq +
    scale_y_continuous(
        breaks = c(0, 0.25, 0.5, 0.75)
    )+
    scale_x_continuous(
        breaks = c(seq(150, 1500, length.out = 4))
    )+
    coord_cartesian(
        xlim = c(150, 1500),
        ylim = c(0, 0.75),
        expand = 0,
        ) +
    ylab(NULL)

# D - GAF PR l2FC vs NURF+BAP l2FC
S7D <- fc_corr.jj(PR_GAF.res, PR_NURFBAP.res)+
  coord_cartesian(
    xlim = c(-8,8), ylim = c(-8,8), expand = FALSE
    )+
  xlab("GAF l2FC vs. LACZ PR")+
  ylab("NURF+BAP l2FC vs. LACZ PR")

S7_all <- (S7A + S7Ba + S7Bb) / (S7Ba + S7Bb + S7D)  +
  plot_layout(guides = "collect")+
  plot_annotation(tag_levels = 'A') &
    theme(plot.tag = element_text(
      family = "Helvetica", face = "bold", size = 12))
```

Figure 3

A - Hsp27 browser shot with RNA-seq RPM on right
B - ATAC-MN metaprofile

```
# A - Hsp27 browser shot with RNA-seq RPM on right
# Getting granges of gene
hsp27.gr <- genes.gr[which(genes.gr$tx_name == "Hsp27-RA")]

# Dividing into PR and GB to highlight pause and GB separately
hsp27_PR.gr <- hsp27.gr
end(hsp27_PR.gr) <- start(hsp27_PR.gr) + 50
start(hsp27_PR.gr) <- start(hsp27_PR.gr) + 25
```

```
hsp27_GB.gr <- hsp27.gr
start(hsp27_GB.gr) <- start(hsp27_GB.gr) + 50

# Adding RNA-seq TPM to GB_labs
GB_labs <- c(
  paste0("LACZ_", as.character(round(mean(
    RNA_normed.tbl[
      RNA_normed.tbl$tx_name == "Hsp27-RA",]$LACZ_RNAseq_Rep1,
    RNA_normed.tbl[
      RNA_normed.tbl$tx_name == "Hsp27-RA",]$LACZ_RNAseq_Rep2
  )))),
  paste0("GAF_", as.character(round(mean(
    RNA_normed.tbl[
      RNA_normed.tbl$tx_name == "Hsp27-RA",]$GAF_RNAseq_Rep1,
    RNA_normed.tbl[
      RNA_normed.tbl$tx_name == "Hsp27-RA",]$GAF_RNAseq_Rep2
  )))),
  paste0("BAP_", as.character(round(mean(
    RNA_normed.tbl[
      RNA_normed.tbl$tx_name == "Hsp27-RA",]$BAP170_RNAseq_Rep1,
    RNA_normed.tbl[
      RNA_normed.tbl$tx_name == "Hsp27-RA",]$BAP170_RNAseq_Rep2
  )))),
  paste0("NURF_", as.character(round(mean(
    RNA_normed.tbl[
      RNA_normed.tbl$tx_name == "Hsp27-RA",]$NURF301_RNAseq_Rep2
  )))),
  paste0("N+B_", as.character(round(mean(
    RNA_normed.tbl[
      RNA_normed.tbl$tx_name == "Hsp27-RA",]$NURF301BAP170_RNAseq_Rep1,
    RNA_normed.tbl[
      RNA_normed.tbl$tx_name == "Hsp27-RA",]$NURF301BAP170_RNAseq_Rep2
  ))))
)

# Plotting (see file browserPlotR.R for more detail)
F3Aa <- browser_plotter.jj(
    hsp27_PR.gr,
    list("PROseq" = PROseq_normed_merged.lst),
    labs = c("", "", "", "", ""),
    scale_bar_size = 10,
    pad_left = 0,
    pad_right = 0,
    binsize = 1,
    bin_FUN = mean)

F3Ab <- browser_plotter.jj(
    hsp27_GB.gr,
    list("PROseq" = PROseq_normed_merged.lst),
    labs = GB_labs,
    scale_bar_size = 500,
    pad_left = 0,
    pad_right = 0,
```

```
    binsize = 1,
    bin_FUN = mean)

# C - ATAC-MN metaprofile
# Getting list of NURF-dependent promoters (FDR 0.01)
PR_NURFres.df <- as.data.frame(PR_NURF.res)
PR_NURFres.df$tx_name <- row.names(PR_NURFres.df)
nurfDepPR.gr <-
    genes.gr[which(
        genes.gr$tx_name %in%
            (PR_NURFres.df %>%
            filter(padj < 0.01 & log2FoldChange > 0))$tx_name
    )]

# Getting counts matrix of ATAC data
nurfDepPR_ATACMNSub.mat <-
    metaSubsample(
        dataset.gr = ATACMN_normed_merged.lst,
        regions.gr = promoters(nurfDepPR.gr, 500, 500),
        binsize = 5,
        first.output.xval = -500,
        expand_ranges = TRUE
    )

# Factoring sample.name column
nurfDepPR_ATACMNSub.mat$sample.name <-
    factor(
        nurfDepPR_ATACMNSub.mat$sample.name,
        levels = names(ATACMN_normed_merged.lst)
    )

# Plotting
F3B <- ggMetaplot(nurfDepPR_ATACMNSub.mat) +
    gg_color_scale_RNAi_ATACMN +
    gg_fill_scale_RNAi_ATACMN +
    scale_x_continuous(
        limits = c(-500, 500),
        expand = c(0,0)
    ) +
    scale_y_continuous(
        limits = c(0, .8),
        expand = c(0,0)
    )+
  ggtitle("ATAC 130-200 bp")

# Laying out panels
F3_layout <- "
BCCA
BCC#
"

F3_all <- F3B + F3Aa + F3Ab +
  plot_layout(guides = "collect", design = F3_layout)+
```

```
  plot_annotation(tag_levels = 'A') &
    theme(plot.tag = element_text(
      family = "Helvetica", face = "bold", size = 12))
```

Figure S8

A - MA plot of GAF vs. PBAP ATAC peaks
B - Venn diagram of GAF and PBAP promoter intersections
C - GAF ChIP-seq of three classes
D - ATAC-seq metaprofiles of three classes
E - PRO-seq pause profiles of three classes
F - motif analysis

```
# A - MA plot of GAF vs. PBAP ATAC peaks
# Getting counts table
ATAC_peaks.df <- data.frame(ATAC_peaks.tbl, row.names = "peak_name")
ATAC_peaks_col_data.df <- data.frame(
  row.names = colnames(ATAC_peaks.df),
  RNAi = c(
    "BAP170",
    "BAP170",
    "GAF",
    "GAF",
    "LACZ",
    "LACZ",
    "NURFBAP",
    "NURFBAP",
    "NURF301",
    "NURF301"
  )
)

# Running DESeq2
ATAC_peaks_GAFvsBAP.res <-
  subset_DESeq.jj(ATAC_peaks.df,
                  ATAC_peaks_col_data.df,
                  scale_facts_ATACDHS.dbl,
                  c(3,4,1,2),
                  c("RNAi", "GAF", "BAP170"),
                  ~RNAi)

# Plotting
S8A <- maplot.jj(ATAC_peaks_GAFvsBAP.res)+
  scale_y_continuous(limits = c(-10, 10),
                     breaks = c(-10,-5,0,5,10),
                     expand= c(0,0))


# E - Venn diagram of GAF and PBAP promoter intersections
# Getting lists of GAF and BAP dependent pause genes
bapDepGenes.chr <- rownames(
  PR_BAP.res[which(PR_BAP.res$padj < 0.01 & PR_BAP.res$log2FoldChange < 0), ])
```

```
gafDepGenes.chr <- rownames(
  PR_GAF.res[which(PR_GAF.res$padj < 0.01 & PR_GAF.res$log2FoldChange < 0), ])

# Defining 3 sets of genes:
# GAF-independent but BAP-dependent
gafIndBapDep.gr <- genes.gr[
  genes.gr$tx_name %in% bapDepGenes.chr & !genes.gr$tx_name %in% gafDepGenes.chr,
]
# GAF- and BAP-dependent
gafDepBapDep.gr <- genes.gr[
  genes.gr$tx_name %in% gafDepGenes.chr & genes.gr$tx_name %in% bapDepGenes.chr
]

# GAF-dependent but BAP-independent
gafDepBapInd.gr <- genes.gr[
  genes.gr$tx_name %in% gafDepGenes.chr & !genes.gr$tx_name %in% bapDepGenes.chr,
]

# Plotting Euler diagram (area-proportional Venn diagram)
S8B <- plot(euler(c("GAF" = length(gafDepBapInd.gr),
                            "BAP" = length(gafIndBapDep.gr),
                            "GAF&BAP" = length(gafDepBapDep.gr)),
             shape = "ellipse"),
             quantities = TRUE)

# H - GAF ChIP-seq of three classes
# Getting signal DF
gafBapDepSorted_GAFChIP.df <-
  data.frame(
    "x" = rep(seq(-497.5, 497.5, 5), 3),
    "signal" = c(
      colMeans(getCountsByPositions(
        ChIP.lst$GAF_ChIPseq,
        promoters(gafDepBapInd.gr, 500, 500),
        binsize = 5,
        expand_ranges = TRUE
        )),
      colMeans(getCountsByPositions(
        ChIP.lst$GAF_ChIPseq,
        promoters(gafDepBapDep.gr, 500, 500),
        binsize = 5,
        expand_ranges = TRUE
        )),
      colMeans(getCountsByPositions(
        ChIP.lst$GAF_ChIPseq,
        promoters(gafIndBapDep.gr, 500, 500),
        binsize = 5,
        expand_ranges = TRUE
        ))
      ),
    "class" = c(
      rep("gafDepBapInd", 200),
      rep("gafDepBapDep", 200),
```

```
      rep("gafIndBapDep", 200)
      )

  )
# Plotting
S8C <- ggplot(gafBapDepSorted_GAFChIP.df,
              aes(x = x, y = signal, color = class))+
  geom_line()+
  ggtheme.jj()+
  scale_x_continuous(expand = c(0,0))+
  scale_y_continuous(limits = c(0, 75), breaks = c(0, 25, 50, 75), expand = c(0,0))+
  ylab("Mean")+
  xlab("Distance from TSS")+
  ggtitle("GAF ChIP-seq")

# D - ATAC-seq metaprofiles of three classes
# Getting counts matrix of ATAC data
gafIndBapDep_ATACDHSSub.mat <-
    metaSubsample(
        dataset.gr = ATACDHS_normed_merged.lst,
        regions.gr = promoters(gafIndBapDep.gr, 500, 500),
        binsize = 1,
        first.output.xval = -500,
        expand_ranges = TRUE
    )

gafDepBapDep_ATACDHSSub.mat <-
    metaSubsample(
        dataset.gr = ATACDHS_normed_merged.lst,
        regions.gr = promoters(gafDepBapDep.gr, 500, 500),
        binsize = 1,
        first.output.xval = -500,
        expand_ranges = TRUE
    )

gafDepBapInd_ATACDHSSub.mat <-
    metaSubsample(
        dataset.gr = ATACDHS_normed_merged.lst,
        regions.gr = promoters(gafDepBapInd.gr, 500, 500),
        binsize = 1,
        first.output.xval = -500,
        expand_ranges = TRUE
    )

# Factoring sample.name column
gafIndBapDep_ATACDHSSub.mat$sample.name <-
    factor(
        gafIndBapDep_ATACDHSSub.mat$sample.name,
        levels = names(ATACDHS_normed_merged.lst)
    )

gafDepBapDep_ATACDHSSub.mat$sample.name <-
    factor(
```

```
        gafDepBapDep_ATACDHSSub.mat$sample.name,
        levels = names(ATACDHS_normed_merged.lst)
    )

gafDepBapInd_ATACDHSSub.mat$sample.name <-
    factor(
        gafDepBapInd_ATACDHSSub.mat$sample.name,
        levels = names(ATACDHS_normed_merged.lst)
    )

# Plotting
S8Da <- ggMetaplot(gafDepBapInd_ATACDHSSub.mat) +
    gg_color_scale_RNAi_ATACDHS +
    gg_fill_scale_RNAi_ATACDHS +
    scale_x_continuous(
        limits = c(-500, 500),
        expand = c(0,0)
    ) +
    scale_y_continuous(
        limits = c(0, 200),
        expand = c(0,0)
    )+
  ggtitle("ATAC < 120 bp")


S8Db <- ggMetaplot(gafDepBapDep_ATACDHSSub.mat) +
    gg_color_scale_RNAi_ATACDHS +
    gg_fill_scale_RNAi_ATACDHS +
    scale_x_continuous(
        limits = c(-500, 500),
        expand = c(0,0)
    ) +
    scale_y_continuous(
        limits = c(0, 200),
        expand = c(0,0)
    )+
  ggtitle("ATAC < 120 bp")

S8Dc <- ggMetaplot(gafIndBapDep_ATACDHSSub.mat) +
    gg_color_scale_RNAi_ATACDHS +
    gg_fill_scale_RNAi_ATACDHS +
    scale_x_continuous(
        limits = c(-500, 500),
        expand = c(0,0)
    ) +
    scale_y_continuous(
        limits = c(0, 200),
        expand = c(0,0)
    )+
  ggtitle("ATAC < 120 bp")

# E - PRO-seq pause profiles of three classes
## Getting counts matrix
```

```
gafIndBapDep_proSeqSubPR.mat <-
     metaSubsample(
        dataset.gr = PROseq_normed_merged.lst,
        regions.gr = promoters(gafIndBapDep.gr, 0, 150),
        binsize = 2,
        first.output.xval = 0,
        expand_ranges = FALSE
    )

gafDepBapDep_proSeqSubPR.mat <-
     metaSubsample(
        dataset.gr = PROseq_normed_merged.lst,
        regions.gr = promoters(gafDepBapDep.gr, 0, 150),
        binsize = 2,
        first.output.xval = 0,
        expand_ranges = FALSE
    )

gafDepBapInd_proSeqSubPR.mat <-
     metaSubsample(
        dataset.gr = PROseq_normed_merged.lst,
        regions.gr = promoters(gafDepBapInd.gr, 0, 150),
        binsize = 2,
        first.output.xval = 0,
        expand_ranges = FALSE
    )

# Filling sample.name column and factoring
gafIndBapDep_proSeqSubPR.mat$sample.name <-
    factor(
        gafIndBapDep_proSeqSubPR.mat$sample.name,
        levels = names(PROseq_normed_merged.lst)
    )

gafDepBapDep_proSeqSubPR.mat$sample.name <-
    factor(
        gafDepBapDep_proSeqSubPR.mat$sample.name,
        levels = names(PROseq_normed_merged.lst)
    )

gafDepBapInd_proSeqSubPR.mat$sample.name <-
    factor(
        gafDepBapInd_proSeqSubPR.mat$sample.name,
        levels = names(PROseq_normed_merged.lst)
    )

# Plotting promoter
S8Ea <- ggMetaplot(gafDepBapInd_proSeqSubPR.mat)+
    gg_color_scale_RNAi_PROseq +
    gg_fill_scale_RNAi_PROseq +
    scale_x_continuous(
        limits = c(0, 150),
        expand = c(0,0),
```

```
        breaks = c(0, 50, 100, 150)
    ) +
    scale_y_continuous(
        breaks = c(0, 25, 50, 75)
    ) +
    coord_cartesian(
        ylim = c(0, 75),
        expand = 0,
        )+
  ggtitle("PRO-seq")



S8Eb <- ggMetaplot(gafDepBapDep_proSeqSubPR.mat)+
    gg_color_scale_RNAi_PROseq +
    gg_fill_scale_RNAi_PROseq +
    scale_x_continuous(
        limits = c(0, 150),
        expand = c(0,0),
        breaks = c(0, 50, 100, 150)
    ) +
    scale_y_continuous(
        breaks = c(0, 25, 50, 75)
    ) +
    coord_cartesian(
        ylim = c(0, 75),
        expand = 0,
        )+
  ggtitle("PRO-seq")

S8Ec <- ggMetaplot(gafIndBapDep_proSeqSubPR.mat)+
    gg_color_scale_RNAi_PROseq +
    gg_fill_scale_RNAi_PROseq +
    scale_x_continuous(
        limits = c(0, 150),
        expand = c(0,0),
        breaks = c(0, 50, 100, 150)
    ) +
    scale_y_continuous(
        breaks = c(0,25, 50, 75)
    ) +
    coord_cartesian(
        ylim = c(0, 75),
        expand = 0,
        )+
  ggtitle("PRO-seq")

# I - motif analysis
# Saving bed files of promoters. I'll take these, use
# bedtools getfasta -s to get stranded sequences, and
# then use DREME to search gafIndBapDep and gafDepBapInd
# vs. gafDepBapDep using DREME, e < 0.001.
# Then input found motifs in TomTom
```

```
export.bed(unique(
  promoters(gafIndBapDep.gr, 500, 0)), "bed/gafIndBapDep.bed")
export.bed(unique(
  promoters(gafDepBapInd.gr, 500, 0)), "bed/gafDepBapInd.bed")
export.bed(unique(
  promoters(gafDepBapDep.gr, 500, 0)), "bed/gafDepBapDep.bed")

# Combining all panels
S8_layout <- "
ABBC
#DEF
#GHI
"

S8_all <- S8A + S8B + S8C + S8Da + S8Db + S8Dc + S8Ea + S8Eb + S8Ec +
  plot_layout(guides = "collect", design = S9_layout)+
  plot_annotation(tag_levels = 'A') &
    theme(plot.tag = element_text(
      family = "Helvetica", face = "bold", size = 12))
```

Figure S9

A - HM of GAF/NURF CUT&RUN at all promoters sorted by GAF signal
B - GB PRO-seq l2FC vs RNA-seq l2FC NURF-RNAi
C - NURF GBupRNAup and GBupRNAdown sorted PI distributions
D - NURF GBupRNAup and GBupRNAdown sorted LACZ RNA-seq distributions
E - NURF GBupRNAup and GBupRNAdown sorted ATACDHS metaprofiles
F - NURF GBupRNAup and GBupRNAdown sorted ATACMN metaprofiles
G - NURF GBupRNAup and GBupRNAdown sorted PRO-seq metaprofiles

```
# A - HM of GAF/NURF CUT&RUN at all promoters sorted by GAF signal
# Getting total GAF signal from TSS-500 to TSS in new genes.gr object
genes_gafCaRSorted.gr <- genes.gr
genes_gafCaRSorted.gr$GAF_CaR_signal <-
  getCountsByRegions(
    CaR.lst$GAF_CUTandRUN,
    promoters(genes_gafCaRSorted.gr, 500, 0),
    expand_ranges = TRUE
  )

# Sorting by GAF CUT&RUN signal
genes_gafCaRSorted.gr <-
  sort(genes_gafCaRSorted.gr, by = ~GAF_CaR_signal, decreasing = TRUE)

# Taking only top quartile
genes_gafCaRSorted.gr <-
  genes_gafCaRSorted.gr[
    genes_gafCaRSorted.gr$GAF_CaR_signal > quantile(
      genes_gafCaRSorted.gr$GAF_CaR_signal)[[4]],]

# Getting signal matrix
genes_gafCaRSorted_CaR.mat <- cbind(
```

```
  getCountsByPositions(
    dataset.gr = CaR.lst$GAF_CUTandRUN,
    regions.gr = promoters(genes_gafCaRSorted.gr, 500, 500),
    binsize = 10,
    expand_ranges = TRUE
  ),
  getCountsByPositions(
    dataset.gr = CaR.lst$NURF_CUTandRUN,
    regions.gr = promoters(genes_gafCaRSorted.gr, 500, 500),
    binsize = 10,
    expand_ranges = TRUE
  )
)

# Plotting Heatmap
col_fun = colorRamp2(c(0, 10, 20), viridis(3))
S9A <- Heatmap(
  genes_gafCaRSorted_CaR.mat,
  cluster_columns = F,
  cluster_rows = F,
  col = col_fun,
  use_raster = F
)

# B - RNA-seq vs PRO-seq GB scatter plot
NURF_PRO_RNA.df <-
  data.frame(
    tx_name = row.names(as.data.frame(GB_NURF.res)),
    PRO_l2FC = as.data.frame(GB_NURF.res)$log2FoldChange,
    PRO_padj = as.data.frame(GB_NURF.res)$padj,
    RNA_l2FC = as.data.frame(RNA_NURF.res)$log2FoldChange,
    RNA_padj = as.data.frame(RNA_NURF.res)$padj
  )

# Eliminating non-significant rows
NURF_PRO_RNA.df <- drop_na(NURF_PRO_RNA.df)
NURF_PRO_RNA.df <-
  NURF_PRO_RNA.df[
    NURF_PRO_RNA.df$PRO_padj < 0.1 & NURF_PRO_RNA.df$RNA_padj < 0.1,
    ]


# Counting number points in each quadrant
num_top_left <-
  nrow(NURF_PRO_RNA.df[
    NURF_PRO_RNA.df$PRO_l2FC < 0 & NURF_PRO_RNA.df$RNA_l2FC > 0,
    ])
num_top_right <-
  nrow(NURF_PRO_RNA.df[
    NURF_PRO_RNA.df$PRO_l2FC > 0 & NURF_PRO_RNA.df$RNA_l2FC > 0,
    ])
num_bottom_right <-
  nrow(NURF_PRO_RNA.df[
```

```
    NURF_PRO_RNA.df$PRO_l2FC > 0 & NURF_PRO_RNA.df$RNA_l2FC < 0,
    ])
num_bottom_left <-
  nrow(NURF_PRO_RNA.df[
    NURF_PRO_RNA.df$PRO_l2FC < 0 & NURF_PRO_RNA.df$RNA_l2FC < 0,
    ])

# Creating annotation df for adding numbers to plot
annotations <- data.frame(
        xpos = c(-3.5,-3.5, 3.5, 3.5),
        ypos =  c(-0.5, 0.5,-0.5, 0.5),
        annotateText = c(
          num_bottom_left,
          num_top_left,
          num_bottom_right,
          num_top_right))

S9B <- ggplot(NURF_PRO_RNA.df, aes(x = PRO_l2FC, y = RNA_l2FC))+
  geom_point(size = 0.5, alpha = 0.75)+
  ggtheme.jj()+
  geom_text(
    data=annotations,
    aes(x=xpos,y=ypos,label=annotateText))+
  geom_vline(xintercept = 0, linetype = "dashed", alpha = 0.75) +
  geom_hline(yintercept = 0, linetype = "dashed", alpha = 0.75) +
  coord_cartesian(xlim = c(-4, 4), ylim = c(-4, 4), expand = FALSE)+
  ggtitle("PRO-seq GB vs. RNA-seq")

# C - NURF GBupRNAup and GBupRNAdoum sorted PI distributions
# Getting lists of genes
NURFupRNAup.gr <-
  genes.gr[
    genes.gr$tx_name %in%
      NURF_PRO_RNA.df[
        NURF_PRO_RNA.df$PRO_l2FC > 0 & NURF_PRO_RNA.df$RNA_l2FC > 0,
        ]$tx_name
    ]

NURFupRNAdown.gr <-
  genes.gr[
    genes.gr$tx_name %in%
      NURF_PRO_RNA.df[
        NURF_PRO_RNA.df$PRO_l2FC > 0 & NURF_PRO_RNA.df$RNA_l2FC < 0,
        ]$tx_name
    ]

# Adding gene length
NURFupRNAup.gr$length <- lengths(NURFupRNAup.gr)
NURFupRNAdown.gr$length <- lengths(NURFupRNAdown.gr)

# Adding pause LACZ PRO-seq signal
NURFupRNAup.gr$pause_signal <-
  getCountsByRegions(
```

```
    PROseq_normed_merged.lst$LACZ_PROseq,
    genebodies(NURFupRNAup.gr, -50, 100, fix.end = "start")
  )

NURFupRNAdown.gr$pause_signal <-
  getCountsByRegions(
    PROseq_normed_merged.lst$LACZ_PROseq,
    genebodies(NURFupRNAdown.gr, -50, 100, fix.end = "start")
  )

# Adding GB LACZ PRO-seq signal
NURFupRNAup.gr$GB_signal <-
  getCountsByRegions(
    PROseq_normed_merged.lst$LACZ_PROseq,
    genebodies(NURFupRNAup.gr, 200, -200)
  )

NURFupRNAdown.gr$GB_signal <-
  getCountsByRegions(
    PROseq_normed_merged.lst$LACZ_PROseq,
    genebodies(NURFupRNAdown.gr, 200, 200)
  )

# Adding norm factor for length normalizing GB signal
NURFupRNAup.gr$length_scaleFact <- 150 / NURFupRNAup.gr$length
NURFupRNAdown.gr$length_scaleFact <- 150 / NURFupRNAdown.gr$length

# Calculating PI
NURFupRNAup.gr$PI <-
  NURFupRNAup.gr$pause_signal /
  (NURFupRNAup.gr$GB_signal *
     NURFupRNAup.gr$length_scaleFact)

NURFupRNAdown.gr$PI <-
  NURFupRNAdown.gr$pause_signal /
  (NURFupRNAdown.gr$GB_signal *
     NURFupRNAdown.gr$length_scaleFact)

# Getting dataframe of PIs
PI.df <- rbind(
  data.frame(class = "NURFupRNAup", PI = NURFupRNAup.gr$PI),
  data.frame(class = "NURFupRNAdown", PI = NURFupRNAdown.gr$PI)
)

S9C <- ggplot(PI.df, aes(x = class, y = PI, fill = class))+
  geom_violin()+
  geom_boxplot(width = 0.2, fill = "white", outlier.shape = NA)+
  stat_compare_means(method = "wilcox.test")+
  ggtheme.jj()+
  scale_fill_manual(values = updown_cols.chr,
                    breaks = c("NURFupRNAup", "NURFupRNAdown"),
                    labels = c("up_up", "up_down")) +
  theme(axis.text.x = element_blank(),
```

```
        axis.ticks.x = element_blank(),
        axis.title.x = element_blank(),
        axis.line.x = element_blank())+
  ylab("log10 Pause Index")+
  scale_y_log10(
    limits = c(1e-2, 1e4),
    breaks = c(1e-2, 1e0, 1e2, 1e4),
    labels = c("-2", "0", "2", "4"),
    expand = c(0,0)
  )


# D - NURF GBupRNAup and GBupRNAdown sorted LACZ RNA-seq distributions
# Adding LACZ RNA-seq signal
NURFupRNAup.gr$RNA_signal <-
  getCountsByRegions(
    RNAseq_normed_merged.lst$LACZ_RNAseq,
    genebodies(NURFupRNAup.gr, -1000, 0, fix.start = "end")
  )

NURFupRNAdown.gr$RNA_signal <-
  getCountsByRegions(
    RNAseq_normed_merged.lst$LACZ_RNAseq,
    genebodies(NURFupRNAdown.gr, -1000, 0, fix.start = "end")
  )

RNA.df <- rbind(
  data.frame(class = "NURFupRNAup", RNA = NURFupRNAup.gr$RNA_signal),
  data.frame(class = "NURFupRNAdown", RNA = NURFupRNAdown.gr$RNA_signal)
)

S9D <- ggplot(RNA.df, aes(x = class, y = RNA, fill = class))+
  geom_violin()+
  geom_boxplot(width = 0.2, fill = "white", outlier.shape = NA)+
  stat_compare_means(method = "wilcox.test")+
  ggtheme.jj()+
  scale_fill_manual(values = updown_cols.chr,
                    breaks = c("NURFupRNAup", "NURFupRNAdown"),
                    labels = c("up_up", "up_down")) +
  theme(axis.text.x = element_blank(),
        axis.ticks.x = element_blank(),
        axis.title.x = element_blank(),
        axis.line.x = element_blank())+
  ylab("log10 RNA-seq")+
  scale_y_log10(
    limits = c(1e0, 1e6),
    breaks = c(1e0, 1e2, 1e4, 1e6),
    labels = c("0", "2", "4", "6"),
    expand = c(0,0)
  )

# E - NURF GBupRNAup and GBupRNAdown sorted ATACDHS metaprofiles
NURFupRNAupATACDHS.df <-
```

```
    metaSubsample(
        dataset.gr = ATACDHS_normed_merged.lst,
        regions.gr = promoters(NURFupRNAup.gr, 500, 500),
        binsize = 1,
        first.output.xval = -500,
        expand_ranges = TRUE
    )

NURFupRNAupATACDHS.df$class <- "up_up"

NURFupRNAdownATACDHS.df <-
    metaSubsample(
        dataset.gr = ATACDHS_normed_merged.lst,
        regions.gr = promoters(NURFupRNAdown.gr, 500, 500),
        binsize = 1,
        first.output.xval = -500,
        expand_ranges = TRUE
    )

NURFupRNAdownATACDHS.df$class <- "up_down"

nurfSorted_ATACDHS.df <- rbind(NURFupRNAupATACDHS.df,
                               NURFupRNAdownATACDHS.df)

S9E <- ggMetaplot(nurfSorted_ATACDHS.df) +
    gg_color_scale_RNAi_ATACDHS +
    gg_fill_scale_RNAi_ATACDHS +
    scale_x_continuous(
        limits = c(-500, 500),
        expand = c(0,0)
    ) +
    scale_y_continuous(
        limits = c(0, 120),
        expand = c(0,0)
    )+
  ggtitle("ATAC < 120 bp")+
  facet_grid(~class)


# F - NURF GBupRNAup and GBupRNAdown sorted ATACMN metaprofiles
NURFupRNAupATACMN.df <-
    metaSubsample(
        dataset.gr = ATACMN_normed_merged.lst,
        regions.gr = promoters(NURFupRNAup.gr, 500, 500),
        binsize = 5,
        first.output.xval = -500,
        expand_ranges = TRUE
    )

NURFupRNAupATACMN.df$class <- "up_up"

NURFupRNAdownATACMN.df <-
    metaSubsample(
```

```
        dataset.gr = ATACMN_normed_merged.lst,
        regions.gr = promoters(NURFupRNAdown.gr, 500, 500),
        binsize = 5,
        first.output.xval = -500,
        expand_ranges = TRUE
    )

NURFupRNAdownATACMN.df$class <- "up_down"

nurfSorted_ATACMN.df <- rbind(NURFupRNAupATACMN.df,
                              NURFupRNAdownATACMN.df)

S9F <- ggMetaplot(nurfSorted_ATACMN.df) +
    gg_color_scale_RNAi_ATACDHS +
    gg_fill_scale_RNAi_ATACDHS +
    scale_x_continuous(
        limits = c(-500, 500),
        expand = c(0,0)
    ) +
    scale_y_continuous(
        limits = c(0, 1.0),
        expand = c(0,0)
    )+
  ggtitle("ATAC 130-200 bp")+
  facet_grid(~class)

# G - NURF GBupRNAup and GBupRNAdown sorted PRO-seq metaprofiles
NURFupRNAup_PR.mat <-
     metaSubsample(
        dataset.gr = PROseq_normed_merged.lst,
        regions.gr = promoters(NURFupRNAup.gr, 0, 150),
        binsize = 2,
        first.output.xval = 0,
        expand_ranges = FALSE
    )

NURFupRNAup_PR.mat$class <- "up_up"

NURFupRNAdown_PR.mat <-
     metaSubsample(
        dataset.gr = PROseq_normed_merged.lst,
        regions.gr = promoters(NURFupRNAdown.gr, 0, 150),
        binsize = 2,
        first.output.xval = 0,
        expand_ranges = FALSE
    )
NURFupRNAdown_PR.mat$class <- "up_down"

NURFupRNAup_GB.mat <-
     metaSubsample(
        dataset.gr = PROseq_normed_merged.lst,
        regions.gr = promoters(NURFupRNAup.gr, 0, 1500),
        binsize = 20,
```

```
        first.output.xval = 0,
        expand_ranges = FALSE
    )
NURFupRNAup_GB.mat$class <- "up_up"

NURFupRNAdown_GB.mat <-
     metaSubsample(
        dataset.gr = PROseq_normed_merged.lst,
        regions.gr = promoters(NURFupRNAdown.gr, 0, 1500),
        binsize = 20,
        first.output.xval = 0,
        expand_ranges = FALSE
    )
NURFupRNAdown_GB.mat$class <- "up_down"

# Filling sample.name column and factoring
NURFupRNAup_PR.mat$sample.name <-
    factor(
        NURFupRNAup_PR.mat$sample.name,
        levels = names(PROseq_normed_merged.lst)
    )

NURFupRNAdown_PR.mat$sample.name <-
    factor(
        NURFupRNAdown_PR.mat$sample.name,
        levels = names(PROseq_normed_merged.lst)
    )

NURFupRNAup_GB.mat$sample.name <-
    factor(
        NURFupRNAup_GB.mat$sample.name,
        levels = names(PROseq_normed_merged.lst)
    )

NURFupRNAdown_GB.mat$sample.name <-
    factor(
        NURFupRNAdown_GB.mat$sample.name,
        levels = names(PROseq_normed_merged.lst)
    )

nurfSorted_PR.mat <- rbind(
  NURFupRNAup_PR.mat, NURFupRNAdown_PR.mat
)

nurfSorted_GB.mat <- rbind(
  NURFupRNAup_GB.mat, NURFupRNAdown_GB.mat
)

# Plotting promoter
S9Ga <- ggMetaplot(nurfSorted_PR.mat)+
    gg_color_scale_RNAi_PROseq +
    gg_fill_scale_RNAi_PROseq +
    scale_x_continuous(
```

```
        limits = c(0, 150),
        expand = c(0,0),
        breaks = c(0, 50, 100, 150)
    ) +
    scale_y_continuous(
        breaks = c(0, 50, 100, 150)
    ) +
    coord_cartesian(
        ylim = c(0, 150),
        expand = FALSE,
        )+
  ggtitle("PRO-seq")+
  facet_grid(class~.)

# Plotting gene body
S9Gb <- ggMetaplot(nurfSorted_GB.mat)+
    gg_color_scale_RNAi_PROseq +
    gg_fill_scale_RNAi_PROseq +
    scale_y_continuous(
        breaks = c(0, 1, 2, 3)
    )+
    scale_x_continuous(
        breaks = c(seq(150, 1500, length.out = 4))
    )+
    coord_cartesian(
        xlim = c(150, 1500),
        ylim = c(0, 3),
        expand = 0,
        ) +
    ylab(NULL)+
  facet_grid(class~.)

S9_BCD <- (plot_spacer() | S9B | S9C | S9D) +
  plot_layout(guides = "collect")+
  plot_annotation(tag_levels = 'A') &
    theme(plot.tag = element_text(
      family = "Helvetica", face = "bold", size = 12))

S9EF <-  (S9E / S9F) + plot_layout(guides = "collect")+
  plot_annotation(tag_levels = 'A') &
    theme(plot.tag = element_text(
      family = "Helvetica", face = "bold", size = 12))

S9G <-  (S9Ga | S9Gb) + plot_layout(guides = "collect")+
  plot_annotation(tag_levels = 'A') &
    theme(plot.tag = element_text(
      family = "Helvetica", face = "bold", size = 12))
```

Supplementary Code 7. Session Info

The output of this code chunk displays all the versions of packages that were used to analyze the data presented in this manuscript

```
sessionInfo()
```

```
## R version 3.6.3 (2020-02-29)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Catalina 10.15.4
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
##  [1] grid      parallel  stats4    stats     graphics  grDevices utils
##  [8] datasets  methods   base
##
## other attached packages:
##  [1] gridExtra_2.3               eulerr_6.0.2
##  [3] tiff_0.1-5                  circlize_0.4.8
##  [5] ComplexHeatmap_2.0.0        plyr_1.8.6
##  [7] patchwork_1.0.0             extrafont_0.17
##  [9] BRGenomics_0.99.26          rtracklayer_1.46.0
## [11] scales_1.1.0                viridis_0.5.1
## [13] viridisLite_0.3.0           ggpubr_0.2.5
## [15] magrittr_1.5                DESeq2_1.26.0
## [17] SummarizedExperiment_1.16.1 DelayedArray_0.12.2
## [19] BiocParallel_1.20.1         matrixStats_0.56.0
## [21] Biobase_2.46.0              GenomicRanges_1.38.0
## [23] GenomeInfoDb_1.22.0         IRanges_2.20.2
## [25] S4Vectors_0.24.3            BiocGenerics_0.32.0
## [27] forcats_0.5.0               stringr_1.4.0
## [29] dplyr_0.8.5                 purrr_0.3.3
## [31] readr_1.3.1                 tidyr_1.0.2
## [33] tibble_2.1.3                ggplot2_3.3.0
## [35] tidyverse_1.3.0
##
## loaded via a namespace (and not attached):
##  [1] colorspace_1.4-1         rjson_0.2.20             ggsignif_0.6.0
##  [4] htmlTable_1.13.3         XVector_0.26.0           GlobalOptions_0.1.1
##  [7] base64enc_0.1-3          fs_1.3.2                 clue_0.3-57
## [10] rstudioapi_0.11          bit64_0.9-7              AnnotationDbi_1.48.0
## [13] fansi_0.4.1              lubridate_1.7.4          xml2_1.2.2
## [16] splines_3.6.3            geneplotter_1.64.0       knitr_1.28
## [19] Formula_1.2-3            jsonlite_1.6.1           Rsamtools_2.2.3
## [22] Rttf2pt1_1.3.8           broom_0.5.5              annotate_1.64.0
## [25] cluster_2.1.0            dbplyr_1.4.2             png_0.1-7
## [28] compiler_3.6.3           httr_1.4.1               backports_1.1.5
```

```
## [31] assertthat_0.2.1         Matrix_1.2-18           cli_2.0.2
## [34] acepack_1.4.1            htmltools_0.4.0         tools_3.6.3
## [37] gtable_0.3.0             glue_1.3.2              GenomeInfoDbData_1.2.2
## [40] Rcpp_1.0.4               cellranger_1.1.0        Biostrings_2.54.0
## [43] vctrs_0.2.4              nlme_3.1-145            extrafontdb_1.0
## [46] xfun_0.12                rvest_0.3.5             lifecycle_0.2.0
## [49] XML_3.99-0.3             zlibbioc_1.32.0         hms_0.5.3
## [52] RColorBrewer_1.1-2       yaml_2.2.1              memoise_1.1.0
## [55] rpart_4.1-15             latticeExtra_0.6-29     stringi_1.4.6
## [58] RSQLite_2.2.0            genefilter_1.68.0       checkmate_2.0.0
## [61] shape_1.4.4              rlang_0.4.5             pkgconfig_2.0.3
## [64] bitops_1.0-6             evaluate_0.14           lattice_0.20-40
## [67] GenomicAlignments_1.22.1 htmlwidgets_1.5.1       bit_1.1-15.2
## [70] tidyselect_1.0.0         R6_2.4.1                generics_0.0.2
## [73] Hmisc_4.4-0              DBI_1.1.0               pillar_1.4.3
## [76] haven_2.2.0              foreign_0.8-76          withr_2.1.2
## [79] survival_3.1-11          RCurl_1.98-1.1          nnet_7.3-13
## [82] modelr_0.1.6             crayon_1.3.4            rmarkdown_2.1
## [85] GetoptLong_0.1.8         jpeg_0.1-8.1            locfit_1.5-9.2
## [88] readxl_1.3.1             data.table_1.12.8       blob_1.2.1
## [91] reprex_0.3.0             digest_0.6.25           xtable_1.8-4
## [94] munsell_0.5.0
```