Analysis of epigenetic signals captured by fragmentation patterns of cell-free DNA

The following sections provide details for the analyses performed in:

Snyder MW, Kircher M, Hill AJ, Daza RM, Shendure J. Cell-free DNA Comprises an In Vivo Nucleosome Footprint that Informs Its Tissues-Of-Origin. Cell. 2016 Jan 14;164(1-2):57-68. doi: 10.1016/j.cell.2015.11.050. PubMed PMID: 26771485

*All scripts and binaries are provided as is, without any warrenty and for use at your own risk. This is not the release of a software package. We are only providing this information and code in addition to a description of methods for making it easier to reproduce our analyses. We are **not** providing any support for these scripts.*

Versions and dependencies

Our scripts largely depend on Python 2, the pysam, bx and numpy libraries. Here a list of versions that we used:

```
python/2.7.3
numpy/1.7.0
pysam/0.7.5
bx-python/0.6.0
```

We were also using R 3.1.0, UCSC binaries for working with bigWig and bigBed files and tools like tabix and samtools from HTSlib.

Please consider using a software management tool like conda to install the respective packages. You can try more recent versions of the packages, but please keep in mind that Python 3 cannot be used to interpret Python 2.7 code.

Samtools version used

Please note that a samtools binary is included with these scripts. Among other things, this samtools binary allows filtering reads based on insert size/read length. This is an early version of the samtools branch released on https://github.com/mpieva/samtools-patched. For samtool calls that are not filtering for read length/insert size, other (more recent) versions of samtools might be used. Please note though that parameters might be named differently and that the ASCII encoding of read filters is also special to the samtools version that we used.

Primary data processing of sequencing data

Barcoded paired-end (PE) sequencing data was split allowing up to one substitution in the barcode sequence. Fragments shorter than or equal to the read length were consensus-called and adapter-trimmed. Remaining consensus single-end reads (SR) and the individual PE reads were aligned to the human reference genome (GRCh37, 1000 Genomes phase 2 technical reference, ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/technical/reference/phase2_reference_assembly_sequence/) using the ALN algorithm in BWA v0.7.10 (Li and Durbin, 2010). PE reads were further processed with BWA SAMPE to resolve ambiguous placement of read pairs or to rescue missing alignments by a more sensitive alignment step around the location of one placed read end. Aligned SR and PE data were stored in BAM format using the samtools API (Li et al., 2009). BAM files for each sample were merged across lanes and sequencing runs. BAM files for each sample were deposited in the NCBI Gene Expression Omnibus (GEO) with accession GSE71378

(http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE71378). Note that GEO is distributing BAM files through SRA (http://www.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?study=SRP061633 and http://www.ncbi.nlm.nih.gov/Traces/study/?acc=SRP061633). *Update:* SRA has recently converted the provided BAM files to SRA format. You can use the SRA tools to convert back. We are providing a copy of the original BAM files here.

Simulated reads and nucleotide frequencies

Aligned sequencing data was simulated (SR if shorter than 45 bp, PE 45 bp otherwise) for all major chromosomes of the human reference (GRC37h). Kmer nucleotide frequencies (k=2 is presented in the manuscript) were determined from real data on both fragment ends and both strand orientations (`referenceKMers.py`), and for the reference genome on both strands (`BAM2FragKMers.py`). The insert size distribution of the real data was extracted for the 1-500 bp range (`BAM_RG_Length.py --noRG`). Reads were simulated procedurally (implemented in `simulate_reads.py`): at each step (i.e., at least once at each genomic coordinate, depending on desired coverage), (1) the strand is randomly chosen, (2) the ratio of the kmer frequency in the real data to that in the reference sequence is used to randomly decide whether the initiating kmer is considered, (3) an length is sampled from the insert size distribution, and (4) the frequency ratio of the terminal kmer is used to randomly decide whether the generated alignment is reported. The simulated coverage was matched to that of the original data after PCR duplicate removal (by adjusting how often sequences are sampled at each position).

```
# Extract kmers (here 2mers) from reference genome
./referenceKMers.py -k 2 -r '' -o grch37_regChroms_2mers.tsv

# Extact the length distribution from the sample BAM, keep only the 1-500bp
range
./BAM_RG_Length.py --noRG -p tmp_outfile BAMFILE.bam
head -n 501 tmp_outfile_ > ${SAMPLE}_lenDist.tsv

# Extract kmers (here 2mers) of left and right read ends from the SAMPLE
file
./BAM2FragKMers.py --kmerLE=2 --kmerRE=2 -v -r 'ALL' BAMFILE.bam --
outfileLE=${SAMPLE}_left_2mer --outfileRE=${SAMPLE}_right_2mer

# Run simulation for each "regular" chromosome (1..22, X, Y) -- should be
run in parallel
for i in $(head -n 24 grch37_1000g_phase2/whole_genome.fa.fai | awk '{ print
$1":1-"$2 }'); do \
  ./simulate_reads.py -v -d ${SAMPLE}_lenDist.tsv --
```

```
fwdKMerGenome=grch37_regChroms_2mers.tsv --
revKMerGenome=grch37_regChroms_2mers.tsv --
fwdPKMers=${SAMPLE}_left_2mer_f.tsv --fwdMKMers=${SAMPLE}_left_2mer_r.tsv --
revPKMers=${SAMPLE}_right_2mer_f.tsv --revMKMers=${SAMPLE}_right_2mer_r.tsv
-s 20  -r \'"$i"\' -o ${SAMPLE}_2mer_sim_chr$( echo $i | cut -f 1 -
d':').bam; \
done

# Combine chromosome files, index and regenerate some stats
./samtools merge -u ${SAMPLE}_chr*.bam | ./samtools sort -
${SAMPLE}_allChrom
./samtools index ${SAMPLE}_allChrom.bam
./samtools flagstat ${SAMPLE}_allChrom.bam > ${SAMPLE}_allChrom.bam_stats
./samtools view -F _2 ${SAMPLE}_allChrom.bam | cut -f 3 | uniq -c >
${SAMPLE}_allChrom.byChromCounts.txt
./BAM2FragmentationPatterns.py -r ALL -o ${SAMPLE}_allChrom.fragpatterns.txt
${SAMPLE}_allChrom.bam
./BAM_RG_Length.py --noRG -p ${SAMPLE}_allChrom.length
${SAMPLE}_allChrom.bam
```

Analysis of nucleotide composition of 167 bp fragments

Fragments with inferred lengths of exactly 167 bp were filtered within samples to remove duplicates. Dinucleotide frequencies were calculated in a strand-aware manner, using a sliding 2 bp window and reference alleles at each position, beginning 50 bp upstream of one fragment endpoint and ending 50 bp downstream of the other endpoint. Observed dinucleotide frequencies at each position were compared to expected dinucleotide frequencies determined from a set of simulated reads reflecting the same cleavage biases calculated in a library-specific manner.

See the folder `nucleotide_composition` for more details.

Coverage, fragment endpoints, and windowed protection scores

Fragment endpoint coordinates were extracted from BAM files with the SAMtools API. Both outer alignment coordinates of PE data were extracted for properly paired reads. Both end coordinates of SR alignments were extracted when PE data was collapsed to SR data by adapter trimming. A fragment's coverage is defined as all positions between the two (inferred) fragment ends, inclusive of endpoints. We define the Windowed Protection Score (WPS) of a window of size k as the number of molecules spanning the

window minus those with an endpoint within the window. We assign the determined WPS to the center of the window. For 35-80 bp fragments (short fraction, S-WPS), k=16; for 120-180 bp fragments (long fraction, L-WPS), k=120. We store coverage, read starts and WPS in gzip-compressed WIG files. We used wigToBigWig from the UCSC tools (http://hgdownload.cse.ucsc.edu/admin/exe/) for converting these to BigWig (bw) for visualization in genome viewers.

Running the extraction of coverage, read starts and WPS for the long fraction:

```
./samtools view -u -m 120 -M 180 BAMFILE.bam $EXTREGION |
./FilterUniqueBAM.py -p | ./extractReadStartsFromBAM2Wig.py -p -r $REGION -w
120 -c COVERAGE.wig.gz -n WPS.wig.gz -s STARTS.wig.gz
```

Running the extraction of coverage, read starts and WPS for the short fraction:

```
./samtools view -u -m 35 -M 80 BAMFILE.bam $EXTREGION | ./FilterUniqueBAM.py
-p | ./extractReadStartsFromBAM2Wig.py -p -r $REGION -w 16 -c
COVERAGE.wig.gz -n WPS.wig.gz -s STARTS.wig.gz
```

Please note the variables `EXTREGION` and `REGION` above. When piping reads from a region, we might miss the forward read of a read pair and the provided scripts usually only extract information from the first read of a pair. Thus, `EXTREGION` should include additional bases around the region (i.e. 200bp or another value that guarantees that all read insert sizes are included; here 180bp and 80bp would be sufficient; e.g. if REGION is 1:1000-2000, EXTREGION should be 1:800-2200).

Nucleosome peak calling

For nuclesome peak calling, the L-WPS is locally adjusted to a running median of zero in 1 kb windows and smoothed using a Savitzky-Golay filter (Savitzky and Golay, 1964) (window size 21, 2nd order polynomial). The L-WPS track is then segmented into above-zero regions (allowing up to 5 consecutive positions below zero). If the resulting region is 50-150 bp, we identify the median L-WPS value of that region and search for the maximum-sum contiguous window above the median. We report the start, end and center coordinates of this window as the “peak,” or local maximum of nucleosome protection. All calculations involving distances between peaks are based on these center coordinates. A score for each peak is determined as the distance between maximum value in the window and the average of the two adjacent L-WPS minima neighboring the

region. If the identified region is 150-450 bp, we apply the same above median contiguous window approach, but only report those windows that are 50-150 bp. For score calculation of multiple windows derived from 150-450 bp regions, we set the neighboring minima to zero. We discard regions <50 bp or >450 bp.

Peak calling is implemented in `callPeaks.py` and expects WIG on STDIN:

```
# Note that EXTREGION should be larger than REGION (by the maximum read
length, i.e. 180) to prevent skipping alignments
./samtools view -u -m 120 -M 180 BAMFILE.bam $EXTREGION |
./FilterUniqueBAM.py -p | ./extractReadStartsFromBAM2Wig.py -p -r $REGION -w
120 -c OFF -s OFF | ./callPeaks.py -s > calls.bed

# or from bigWig (http://hgdownload.cse.ucsc.edu/admin/exe/) and save as
block-gzip compressed (http://www.htslib.org/doc/tabix.html) BED file:

bigWigToWig -chrom=chr1 -start=12000000 -end=13000000 ${SAMPLE}.bw
/dev/stdout | ./callPeaks.py -s | bgzip -c > calls.bed.gz
```

Again note the variables `EXTREGION` and `REGION` above. When piping reads from a region, we might miss the forward read of a read pair and the provided scripts usually only extract information from the first read of a pair. Thus, `EXTREGION` should include additional bases around the region (i.e. 200bp or another value that guarantees that all read insert sizes are included; here 180bp would be sufficient; e.g. if REGION is 1:1000-2000, EXTREGION should be 1:800-2200).

Bed files can be converted to bigBed using the above mentioned UCSC tools. We uploaded bigBed (bb) files of our nucleosome calls to GEO for BH01, IH01, IH02, CH01, and CA01.

Analysis of DHS sites

DHS peaks for 349 primary tissue and cell line samples were downloaded from http://www.uwencode.org/proj/Science_Maurano_Humbert_et_al/data/all_fdr0.05_hot.tgz. Samples derived from fetal tissues (233 of 349) were removed from the analysis as they behaved inconsistently within tissue type, possibly because of unequal representation of cell types within each tissue sample. 116 DHS callsets from a variety of cell lineages were retained for analysis. For the midpoint of each DHS peak in a particular set, the nearest upstream and downstream calls in the CH01 callset were identified, and the distance between the centers of those two calls was calculated. The distribution of all such distances was visualized for each DHS peak callset using a smoothed density estimate calculated for distances between 0 and 500 bp.

See the folder `DHS` for more details and supporting files.

Analysis of A/B compartments

We downloaded A/B compartment calls with 100 kb resolution from GSE63525, specifically the file `GSE63525_GM12878_subcompartments.bed.gz`. A/B segmentation calls were compared to peak spacing within equivalent 100 kb genomic windows. In a separate analysis, peak density was compared to windowed GC content in 10 kb bins.

See the folder `peak_density` for more details and supporting files.

Annotation of TFBSs and genomic features

We downloaded clustered FIMO (motif-based) intervals (Grant et al., 2011; Maurano et al., 2012, http://www.uwencode.org/proj/Maurano_et_al_func_var/) defining a set of computationally predicted TFBSs (`hg19.taipale.1e-4.starch`). For a subset of clustered TFs (AP-2-2, AP-2, CTCF_Core-2, E2F-2, EBF1, Ebox-CACCTG, Ebox, ESR1, ETS, MAFK, MEF2A-2, MEF2A, MYC-MAX, PAX5-2, RUNX2, RUNX-AML, STAF-2, TCF-LEF, YY1), we retained only predicted TFBSs that overlap with ENCODE ChIP-seq peaks (TfbsClusteredV3 set downloaded from http://hgdownload.cse.ucsc.edu/goldenPath/hg19/encodeDCC/wgEncodeRegTfbsClustered/). Details on download and basic processing of these data sets are available in `data/Maurano_et_al_func_var/README` and `data/ENCODE_TfbsClusteredV3/README`.

Genomic coordinates of transcription start sites, transcription end sites, start codons, and splice donor and acceptor sites were obtained from Ensembl Build version 75 (ftp://ftp.ensembl.org/pub/release-75/gtf/homo_sapiens/Homo_sapiens.GRCh37.75.gtf.gz, details are available in `data/Ensembl_v75/README`).

Filtering active CTCF sites

CTCF sites first included clustered FIMO binding site predictions (described above). This set was intersected with ENCODE ChIP-seq peaks (TfbsClusteredV3, described above), and then further intersected with a set of CTCF binding sites experimentally observed to be active across 19 tissues (Wang et al., 2012; details on the download of this data set are available in `data/Wang_et_al_CTCF/README`), to produce three increasingly stringent sets. For each CTCF site, distances between each of 20 flanking nucleosomes (10 upstream and 10 downstream) were calculated.

The mean S-WPS and L-WPS at each position relative to the center of the CTCF binding motifs were calculated within bins defined by spacing between -1 and +1 nucleosomes (>160 bp, 161-200 bp, 201-230 bp, 231-270 bp, 271-420 bp, 421-460 bp, and >460 bp). In figures, CTCF binding sites are shifted such that the zero coordinate on the x-axis is the center of its 52 bp binding footprint (Ong and Corces, 2014).

Overlaying WPS at aligned genomic features

WPS values for a specific data set (in most cases CH01) and the corresponding simulation were extracted for each position in a 5 kb window around the start coordinate of each TFBS, and was aggregated within each TF cluster. The mean WPS of the first and last 500 bp (which is predominantly flat and represents a mean offset) of the 5 kb window was subtracted from the original WPS at each position. For L-WPS only, a sliding window mean is calculated using a 200 bp window and subtracted from the original signal. Finally, the corrected WPS profile for the simulation is subtracted from the corrected WPS profile of the data set to correct for signal that is a product of fragment length and ligation bias. This final profile is plotted and termed the Adjusted WPS.

We provide Python and R scripts for extracting and plotting WPS overlays from block-gzip compressed WIG files. Such files are generated by `extractReadStartsFromBAM2Wig.py` or `normalizeWPSwigs.py` and need to be tabix-indexed to retrieve specific genomic regions using the tabix routines of pysam.

###Motif-based TF binding site predictions

We outline how S-WPS and L-WPS plots were generated from clustered FIMO (motif-based) sites in a README in `WPS_overlays/Maurano_et_al_TFclusters`.

###Motif-based TF binding sites overlapped with ENCODE ChIP-seq peaks

Please find a README in `WPS_overlays/Maurano_et_al_plus_ChIP` which outlines how S-WPS and L-WPS plots were generated for a subset of clustered FIMO (motif-based) sites overlapped with ENCODE ChIP-seq peaks of AP-2-2, AP-2, CTCF_Core-2, E2F-2, EBF1, Ebox-CACCTG, Ebox, ESR1, ETS, MAFK, MEF2A-2, MEF2A, MYC-MAX, PAX5-2, RUNX2, RUNX-AML, STAF-2, TCF-LEF, and YY1.

###Active CTCF from ChIP-seq in 19 cell-lines

`WPS_overlays/CTCF_19CellLines` has a README file which outlines how S-WPS and L-WPS plots were generated for this subset of CTCF sites.

###Adjusted WPS around genic features

A README describing how the adjusted WPS was extracted and overlayed around TSS, TSE, splice acceptor/donor, and start/stop codon can be found in `WPS_overlays/genicFeatures`.

###Adjusted WPS around genic features in five NB-4 expression bins

The file `WPS_overlays/genicFeatures_by_expression/README` describes how adjusted WPS was extracted and overlayed around TSS, TSE, splice acceptor/donor, and start/stop codon for the five NB-4 expression bins and how the genes in each bin were defined.

Gene expression analysis

###Human Protein Atlas

FPKM gene expression (GE) values measured for 20,344 Ensembl gene identifiers in 44 human cell lines and 32 primary tissues by the Human Protein Atlas (Uhlén et al., 2015) was downloaded from http://www.proteinatlas.org/download/rna.csv.zip. Genes with 3 or more non-zero expression values were retained (n=19,378 genes). The GE data set is provided with one decimal precision for the FPKM values. Thus, a zero GE value (0.0) indicates expression in the interval [0, 0.05) Unless otherwise noted, we set the minimum GE value to 0.04 FPKM before log2-transformation..

###Fast Fourier transformation (FFT) and smoothing of trajectories

We use parameters to smooth (3 bp Daniell smoother; moving average giving half weight to the end values) and de-trend the data (i.e. subtract the mean of the series and remove a linear trend). A recursive time series filter implemented in R was used to remove high frequency variation from trajectories. 24 filter frequencies (1/seq(5,100,4)) were used, and

the first 24 values of the trajectory were taken as init values. The 24-value shift in the resulting trajectories was corrected by repeating the last 24 values of the trajectory.

###FFT intensity correlation with expression

L-WPS was used to calculate periodograms of genomic regions using Fast Fourier Transform (FFT, spec.pgram in R) with frequencies between 1/500 and 1/100 bases. Intensity values for the 120-280 bp frequency range were determined from smooth FFT periodograms. S-shaped Pearson correlation between GE values and FFT intensities was observed around the major inter-nucleosome distance peak, along with a pronounced negative correlation in the 193-199 bp frequency range. The mean intensity in this frequency range was correlated with the average intensity with log2-transformed GE values for downstream analysis.

###Running the analysis

```
zcat transcriptAnno-GRCh37.75.tsv.gz | awk 'BEGIN{ FS="\t"; OFS="\t" }{ if
($5 == "+") { print $1,$2,$3-10000,$3,$5 } else { print $1,$2,$4,$4+10000,$5
} }' > transcriptAnno-GRCh37.75.upstream.tsv
zcat transcriptAnno-GRCh37.75.tsv.gz | awk 'BEGIN{ FS="\t"; OFS="\t" }{ if
($5 == "+") { print $1,$2,$4,$4+10000,$5 } else { print $1,$2,$3-10000,$3,$5
} }' > transcriptAnno-GRCh37.75.downstream.tsv
zcat transcriptAnno-GRCh37.75.tsv.gz | awk 'BEGIN{ FS="\t"; OFS="\t" }{ if
($5 == "+") { print $1,$2,$3-1,$3-1+10000,$5 } else { print $1,$2,$4-1-
10000,$4-1,$5 } }' > transcriptAnno-GRCh37.75.body.tsv

# Extract counts for a sample
mkdir -p body/$SAMPLE
./extractReadStartsFromBAM_Region_WPS.py --minInsert=120 --maxInsert=180 -i
transcriptAnno-GRCh37.75.body.tsv -o 'body/$SAMPLE/block_%s.tsv.gz'
BAMFILE.bam

# Run FFT & convert to summary tbl
mkdir -p /tmp/body/$SAMPLE/fft
( cd body/$SAMPLE/counts; ls block_*.tsv.gz ) | xargs -n 500 Rscript
fft_path.R body/$SAMPLE/counts /tmp/body/$SAMPLE/fft
mkdir -p body/fft_summaries/
./convert_files.py -a transcriptAnno-GRCh37.75.body.tsv -t /tmp/ -r  -p body
-i $SAMPLE
rm -fR /tmp/body/$SAMPLE/fft

# Correlate the intensities with the expression data and generate PDFs in R
R --vanilla --quiet < plots.R
```