Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

Report for Softwareproject

Integrative analysis of next generation sequencing data: binding events of pluripotency transcription factors and histone modification pattern across the genome in MEF and ESC

Nhu Quang Vu, Sonja Batke and Sofya Laskina

AG Functional Genomics, Max Plank Institut for molecular genetics, Ihnestrasse 63-73, 14195 Berlin, Germany.

* Alena van Bömmel and Robert Schöpflin.

Abstract

Motivation: Understanding the mechanical functioning in the reprogramming process of a cell to pluripotency with the usage of transcription factors Oct4, Sox2, Klf4, and cMyc in somatic cells is crucial, as the genome-wide inactivation of the somatic program is initiated at this stage. We used data of RNA-Seq, ATAC-Seq and ChIP-Seq experiments of different TFs and histone modifications (HMs) obtained from the research of Chronis *et al.* (2017). We analyzed and compared the data of ESCs and mouse embryonic fibroblasts (MEFs) to gain a better understanding of the process that leads to pluripotency.
Results: We built a model which uses HMs to predict the level gene expression in somatic and pluripotent cells. Based on that model we identified that H3K9ac and H3K27ac seem to play a key role in the reprogramming. Our results also show that the binding sites for Klf4 and cMyc change from MEF to ESC, which was also demonstrated by Chronis *et al.* (2017).
Availability: The data from the ATAC-seq, Chip-seq and RNA-seq experiments can be found in the GEO database under accession number GSE90895.
Supplementary information: All data is available at Supplementary Material

1 Introduction

Embryonic stem cells (ESC) are heavily researched because of their potential in clinical therapy. This potential is due to the pluripotency character of the cells. By overexpressing the transcription factors (TFs) Oct4, Sox2, Klf4 and cMyc (OSKM) it is possible to induce reprogramming of somatic cells to pluripotent cells (Takahashi and Yamanaka, 2006).

TFs can bind with different parts of the genome to activate a gene. They either bind with enhancers, which can be far away from the gene or they bind with promotors, which are in the immediate area of the transcription start site (TSS). But their binding is dependent on whether the chromatin is open or not, which in turn is influenced by histone modifications (Lawrence *et al.*, 2016).

In this research we use data of MEF and ESC to find the differences and their causes in gene expression. Since MEF and ESC are the two stages which are most different to each other, our results can show the general divergence between the cells.

2 Methods

2.1 Getting data

Experiment data of ChIP-Seq, ATAC-Seq and RNA-Seq of the associated cell line was downloaded in .sra format from the NCBI web site[1]. Locally stored files were then rewritten into .fastq files using *fastq-dump* function in SRA-Toolkit[2].

[1] https://www.ncbi.nlm.nih.gov/sra/

[2] https://www.ncbi.nlm.nih.gov/sra/docs/toolkitsoft/

2.2 Univariate analysis

2.2.1 Quality analysis

Quality of obtained read files was checked with FastQC software (Andrews, 2010).

2.2.2 Aligning to reference genome

Reads from ChIP-Seq and ATAC-Seq experiments were mapped to mouse genome (mm9) with Bowtie2 software (B.Langmead, 2007). Bowtie2 is a memory efficient tool for aligning Next Generation Sequencing data using FM-Index. Only those mapped reads were retained, that had a mapping quality of 10 or more and less than three mismatches. Duplications in read files were eliminated. Mapped reads were translated into coverage track files (.bigWig) with *bamCoverage* of deeptools software (Ramírez *et al.*, 2016). Reads were extended by 200bp. Number of reads was calculated for 25bp bins and then normalized by Reads Per Kilobase per Million mapped reads (RPKM).

RNA-Seq data was aligned to mm9 with STAR 2.4.0.1 software (Dobin *et al.*, 2013) obtaining unique reads with 2 or less mismatches and normalized signal by Reads Per Million. For each read file of the RNA-seq experiment a table with read counts per gene was produced with STAR as well. Wiggle files were converted to .bigWig files with *wigToBigWig* of UCSC Tools (Robert M. Kuhn and David Haussler, and W. James Kent, 2013). As the chromosome names of UCSC differ from those of STAR, an own Python code was written to ignore rows with appearances of noncanonical chromosomes.

2.3 Genomic features and overlap analysis

2.3.1 Peak Calling with MACS2

We used MACS2 software v.2.1.2 (Zhang *et al.*, 2008) to call peaks of our ChIP-Seq and ATAC-Seq data using a bandwidth of 150bp. We set the q-val cutoff to <0.005, the same as in Chronis *et al.* (2017), but we didn't change the default Mfold range [5-50], since the results only differed minimally. As control data we used the WCE ChIP-Seq data for the TFs, epigenetic regulators and for H3K79me2, H3K9me3, H3.3 and H3. MNase ChIP-Seq data was used as control for the remaining histone modifications. For the ChIP-Seq data of H3K9me3, we used broad peak calling because this histone modification doesn't produce narrow peaks.

We have later visualized the peaks, mapping and RNA-Seq data in IGV software v.2.5.0 (Robinson *et al.*, 2011; Thorvaldsdóttir *et al.*, 2013) to look at target genes of different transcription factors such as the gene Ccnb1, also known as Cyclin B1, which is a target gene for cMyc (Menssen and Hermeking, 2002).

2.3.2 Correlation between different histone modifications and transcription factors

For the quality control of our .bam files we were using deepTools *plotFingerprint* and deeptools *plotCorrelation*. One plot with fingerprints was done for all transcription factors and one for all histone modifications in both stages. Before using *plotCorrelation* we generated a matrix from all .bam files using *multiBamSummary*. With *plotCorrelation* we produced a heatmap with the Pearson correlation method and removed outliers for both stages.

2.4 Differential analysis

2.4.1 Analysis of TF peaks

In this step we used the .summits file of our peak calling results to create unified peaks that are 200bp long. With the sub-command *intersect* of the BEDTools software v.2.27.1 (Quinlan and Hall, 2010) and the sub-command *merge* of the BEDOPS software v.2.4.35 (Reynolds *et al.*, 2012), we were able to create files, which were used for the analysis of common peaks between MEFs and ESCs and common peaks between OSKM.

Using promoter regions[3], which we defined as 2000bp upstream of a annotated transcription start site (TSS) of RefSeq genes, we analyzed how many peaks of OSKM were located in promoters.

The results of the analysis were visualized with R (R Development Core Team, 2008) and the package *VennDiagramms* (Chen and Boutros, 2011).

2.4.2 Visualization

The number of significant peaks for both MEF and ESC were visualized with own R script.

Converted bigWig files and bed files with peaks were visualized with IGV. Also for both cell lines heatmaps were produced with deepTools software. First, a table with scores per genome was produced with *computeMatrix*, then a plot itself with *plotHeatmap*. As region files we used a file containing gene scores. This was obtained by putting together a promoter file containing gene names and their position with read table of paired-end RNA-seq data. This file then was divided into files with score above and below 500 and also separated by strand. Signal distribution was calculated relative to center of genomic region, which was set to 2kb up- and downstream, .bigWig files of MEF with associated histone modifications were used as scores. Only modifications with signal were retained in plot.

2.5 Prediction model

We wrote a R-script, which built a multivariate linear regression that models the gene expression of protein coding genes in dependency to the histone modifications. We started by extracting all protein coding genes and their promoters, here defined as 2000bp upstream and 500bp downstream of the TSS, from the ENSEMBL archive of may 2012. Then we counted the number of reads in the promoter regions of each histone modification. The resulting pseudocounts and the genecounts of the RNA-Seq data were log-normalized. To evaluate our model we need to split our data in half into a training and test set. Given these sets we now build the model from our training data and evaluate it with the test data by correlating the predicted expression to the expression measured in the test data.

For this script we used the following packages: biomaRT (Durinck *et al.*, 2009; Brazma *et al.*, 2005), GenomicRanges (Lawrence *et al.*, 2013), GenomicFeatures (Lawrence *et al.*, 2013), bamsignals (Mammana and Helmuth, 2016) and DESeq2 (Love *et al.*, 2014).

3 Results

3.1 Univariate analysis

3.1.1 Data quality

Downloaded .fastq files were overall of a good quality with some exceptions, which had poor per sequence and per base quality, some overrepresented sequences, duplicates,etc. As reads were filtered when aligning to the genome, we skipped further formatting of .fastq files.

[3] `http://hgdownload.soe.ucsc.edu/goldenPath/mm9/bigZips/`

3.1.2 Aligning to reference genome

Reads were mapped with high quality of $> 70\%$ with a few exceptions of about 50%. The table with percentages of mapping is shown in Figure 1.

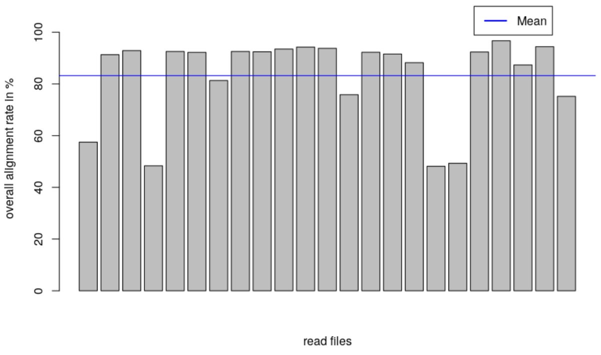


Fig. 1. Percentages of aligned reads for ATAC-Seq and ChIP-Seq experiments of MEF.

3.2 Genomic features and overlap analysis

3.2.1 Peak Calling

Figure 2shows that the majority of significant peaks, with an average length of 432bp, are histone modifications. This is not only because there are more histone modifications than transcription factors, but the absolute numbers are higher in general (Supplementary table S1). In ESCs it is apparent that the amount of significant peaks for the transcription factor cMyc is very low in comparison to the other transcription factors.

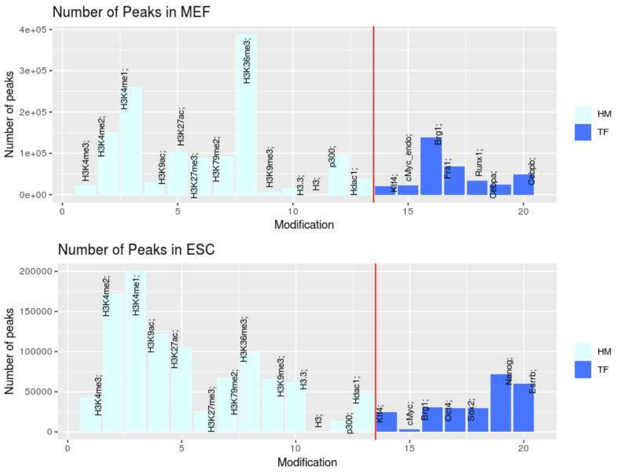


Fig. 2. Number of significant peaks in MEF and ESC with corresponding histone modifications or transcription factors.

3.2.2 Correlation between different histone modifications and transcription factors

Correlation between the different histone modifications and transcription factors from the Chip-Seq and ATAC-Seq experiments in the MEF's and ESC's is shown in Supplementary figure S1. In the MEF heatmap all experiments are very highly correlated, except the ATAC-Seq experiments and H3K79me2. In ESC KLF4, cMyc, p300 and H3K9me3 are very highly correlated, as well as H3K9ac and H3K4me3. Also Sox2, Oct4, H3K27ac and H3K4me1 have a high correlation. Once more ATAC-seq has a low correlation to the other experiments.

Figure S2 shows the fingerprints of all histone modifications and transcription factors in MEF and ESC. Concerning the transcription factors in both stages, the differences between control and signal are less clear than in the plots with the histone modification. There is a higher difference between control and most histone modifications, especially in H3K79me2 and H3K36me3 in the MEF and H3K4me3, H3K4me2 and H3K9ac in the ESC. Nevertheless, we can assume an acceptable quality from the plots.

3.3 Differential analysis

3.3.1 Analysis of TF peaks

The venndiagram in Figure 3 illustrates that Oct4 and Sox2 share many binding sites in ESCs with each other. Klf4 shares a few binding sites with Oct4 and Sox2 and cMyc barely shares any binding site with another transcription factor, but this may be a result of the small amount of significant peaks for cMyc (Supplementary figure S4).

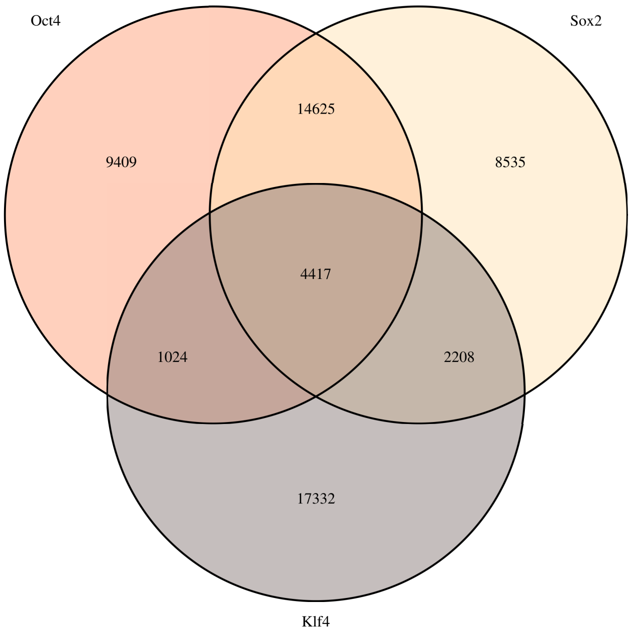


Fig. 3. Number of unique and shared Oct4, Klf4 and Sox2 peaks in ESC

cMyc binding sites are located primarily in promoter regions for both stages. We assume that the other binding sites are located in enhancers. The binding sites for Klf4 however change between the stages. In MEF there are more binding sites in promoters than in enhancers, but in ESC the majority of binding sites are located in enhancers. Distribution of those signals according to their locus is shown in Figure 4.

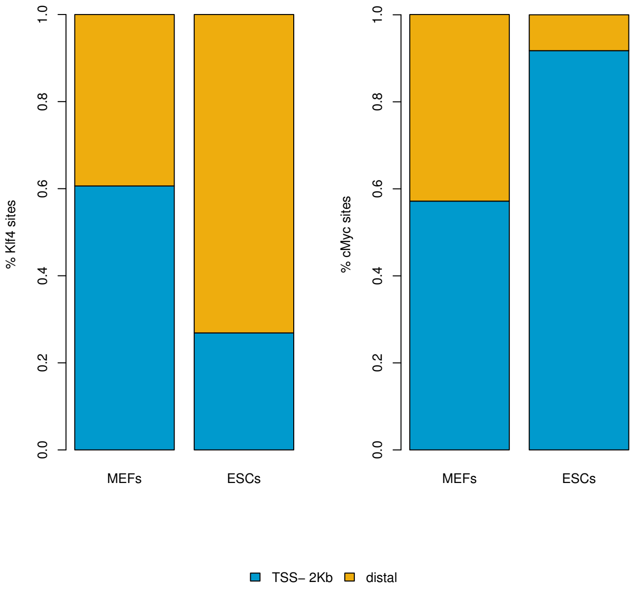


Fig. 4. Klf4 and cMyc binding in promoter region and outside in MEF and ESC

3.3.2 Visualization

Heatmaps on Figure S5 show that most signals for MEF come from H3K4me3, H3K4me2, H3K27ac, H3K79me2 modifications, as the color of the those reads become blue. There is also a reverse dependence between acetylation on 9th and 27th Lysin.

cMyc binds in both MEF and ESC to the promoter region of the target gene Ccnb1. The peaks however differ in size. In ESC the signal is stronger and wider than the signal in MEF. This is also reflected in the RNA-Seq data. The signal in ESC is stronger than the signal in MEF (Supplementary Figure S2).

3.4 Prediction model

The model derived from the histone modifications in MEF is fairly well correlated to the gene expression with r = 0.65 and a p-value below 2.2e-16. (Table 1) H3K27ac, H3K9ac, H3K4me3 seem to be the most important modifications since their absolute values are the highest. We also used this model to predict the gene expression of the test data set. The correlation coefficient between the predicted and true expression is 0.7988768.

We did the same process for histone modifications in ESC and derived a model with r = 0.6997, p-value below 2.2e-16 and coefficients that are listed in Table S2. In this model the histone modification H3K27me3 has one of the highest absolute values, unlike in the MEF model. The predicted and true expression have a correlation coefficient of 0.7376995, which is just a bit worse than the MEF model.

Table 1. β coefficients of the multivariate regression for MEF

modification	value	p-value
(Intercept)	-1.03583	6.36e-11
H3K4me3	1.09168	< 2e-16
H3K4me2	0.64727	< 2e-16
H3K4me1	0.18478	0.00665
H3K9ac	-1.26799	< 2e-16
H3K27ac	1.40176	< 2e-16
H3K27me3	-0.27458	4.80e-08
H3K79me2	0.47766	< 2e-16
H3K36me3	0.06039	0.20053
H3K9me3	-0.89854	< 2e-16

4 Discussion

Based on our plots we can sum up that about 60% of both transcription factors Klf4 and cMyc bind promoter region in MEF and about 40% bind enhancer, when in ESC Klf4 primarily binds enhancer and cMyc regulates transcription by binding promoter region.

In ESC we have seen, that most peaks in Oct4 and Sox2 are shared, which suggest their tandem work. However, Klf4 also shared great number of peaks with this TFs, so it may also be involved in cooperative performance.

The number of significant peaks of H3K4me3, H3K9ac and H3K9me3 rise drastically from MEF to ESC, although the number of significant peaks for all HMs decrease. H3K4me3 and H3K9ac both activate transcription and are important in the MEF model, whereas H3K9me3 is a repressive mark (Lawrence *et al.*, 2016). This suggests that H3K4me3 and H3K9ac mark many genes which are expressed in ESC and H3K9ac represses those genes, which were previously expressed in MEF but aren't in ESC.

The HMs H3K4me3, H3K9ac, and H3K27ac have high values in both the multivariate regression for MEF and ESC. In the MEF model all of these coefficients are positive, which would mean that they increase the level of gene expression. In the ESC model the β coefficient for H3K9ac is negative which contradicts the other model. However these HMs are all correlated with active transcription (Lawrence *et al.*, 2016; Tie *et al.*, 2009), so the ESC model must be faulty somehow.

Our heatmaps showed an expected pattern (K.Barth and Imhof, 2010) of epigenetic regulation. This proves again, that HMs H3K4me3, H3K9ac, and H3K27ac correlate with high level of transcription regulation, although signal of H3K9ac is much stronger in ESC than in MEF and signal of H3K27ac is stronger in MEF. This proposes that acetylation of different Lysins is needed for gene expression in MEF and ESC cell lines.

Acknowledgements

We would like to thank Alena van Bömmel and Robert Schöpflin for their support and helpful advice.

References

Andrews, S. (2010). FastQC: a quality control tool for high throughput sequence data.

B.Langmead, S. (2007). Fast gapped-read alignment with Bowtie 2. *Nature Methods*, **9**, 357–359.

Brazma, A. *et al.* (2005). BioMart and Bioconductor: a powerful link between biological databases and microarray data analysis. *Bioinformatics*, **21**(16), 3439–3440.

Chen, H. and Boutros, P. C. (2011). Venndiagram: a package for the generation of highly-customizable venn and euler diagrams in r. *BMC Bioinformatics*, **12**(1), 35.

Chronis, C. *et al.* (2017). Cooperative binding of transcription factors orchestrates reprogramming. *Cell*, **168**(3), 442 – 459.e20.

Dobin, A. *et al.* (2013). STAR: ultrafast universal RNA-seq aligner. *Bioinformatics*, **29**, 15–21.

Durinck, S. *et al.* (2009). Mapping identifiers for the integration of genomic datasets with the r/bioconductor package biomart. *Nature Protocols*, **4**, 1184 EP –.

K.Barth, T. and Imhof, A. (2010). Fast signals and slow marks: the dynamics of histone modifications. *Trends in Biochemical sciences*, **35**, 618–626.

Lawrence, M. *et al.* (2013). Software for computing and annotating genomic ranges. *PLOS Computational Biology*, **9**(8), 1–10.

Lawrence, M. *et al.* (2016). Lateral thinking: How histone modifications regulate gene expression. *Trends in Genetics*, **32**(1), 42 – 56.

Love, M. I. *et al.* (2014). Moderated estimation of fold change and dispersion for rna-seq data with deseq2. *Genome Biology*, **15**(12), 550.

Mammana, A. and Helmuth, J. (2016). bamsignals: Extract read count signals from bam files. *R package version*, **1**(0).

Menssen, A. and Hermeking, H. (2002). Characterization of the c-myc-regulated transcriptome by sage: Identification and analysis of c-myc target genes. *Proceedings of the National Academy of Sciences*, **99**(9), 6274–6279.

Quinlan, A. R. and Hall, I. M. (2010). BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, **26**(6), 841–842.

R Development Core Team (2008). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

Ramírez *et al.* (2016). deepTools2: A next Generation Web Server for Deep-Sequencing Data Analysis. *Nucleic Acids Research*.

Reynolds, A. P. *et al.* (2012). BEDOPS: high-performance genomic feature operations. *Bioinformatics*, **28**(14), 1919–1920.

Robert M. Kuhn and David Haussler, and W. James Kent (2013). *The UCSC genome browser and associated tools*.

Robinson, J. T. *et al.* (2011). Integrative Genomics Viewer. *Nature Biotechnology*, **29**, 24–26.

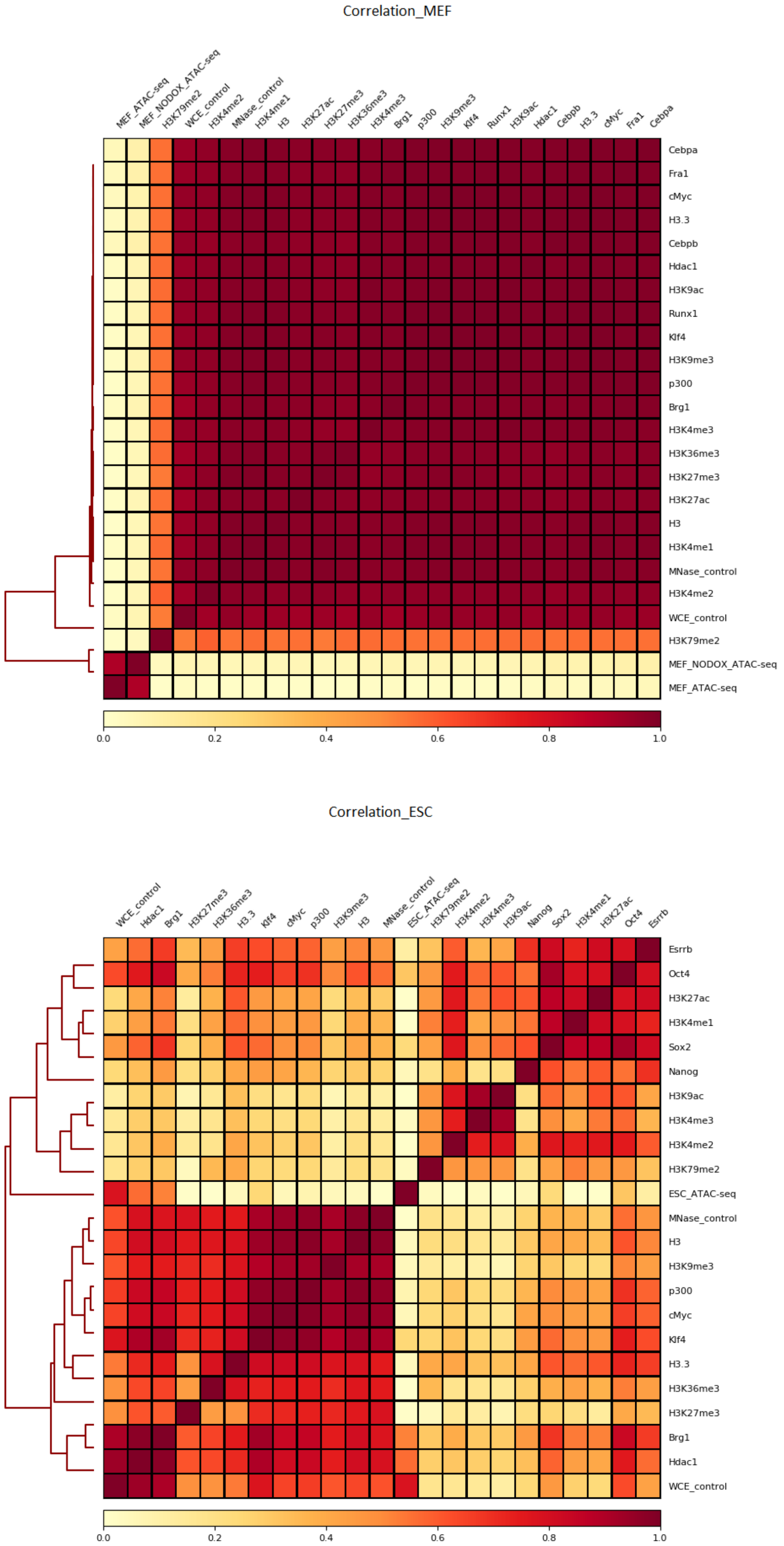
Takahashi, K. and Yamanaka, S. (2006). Induction of pluripotent stem cells from mouse embryonic and adult fibroblast cultures by defined factors. *Cell*, **126**(4), 663 – 676.

Thorvaldsdóttir, H. *et al.* (2013). Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration. *Briefings in Bioinformatics*, **14**, 178–192.

Tie, F. *et al.* (2009). Cbp-mediated acetylation of histone h3 lysine 27 antagonizes drosophila polycomb silencing. *Development*, **136**(18), 3131–3141.

Zhang, Y. *et al.* (2008). Model-based analysis of ChIP-Seq (MACS). *Genome Biol.*, **9**(9), R137.

Supplementary Material



Correlation_MEF

Correlation_ESC

Fig. S1. Correlation of histone modifications and transcription factors in MEF and ESC

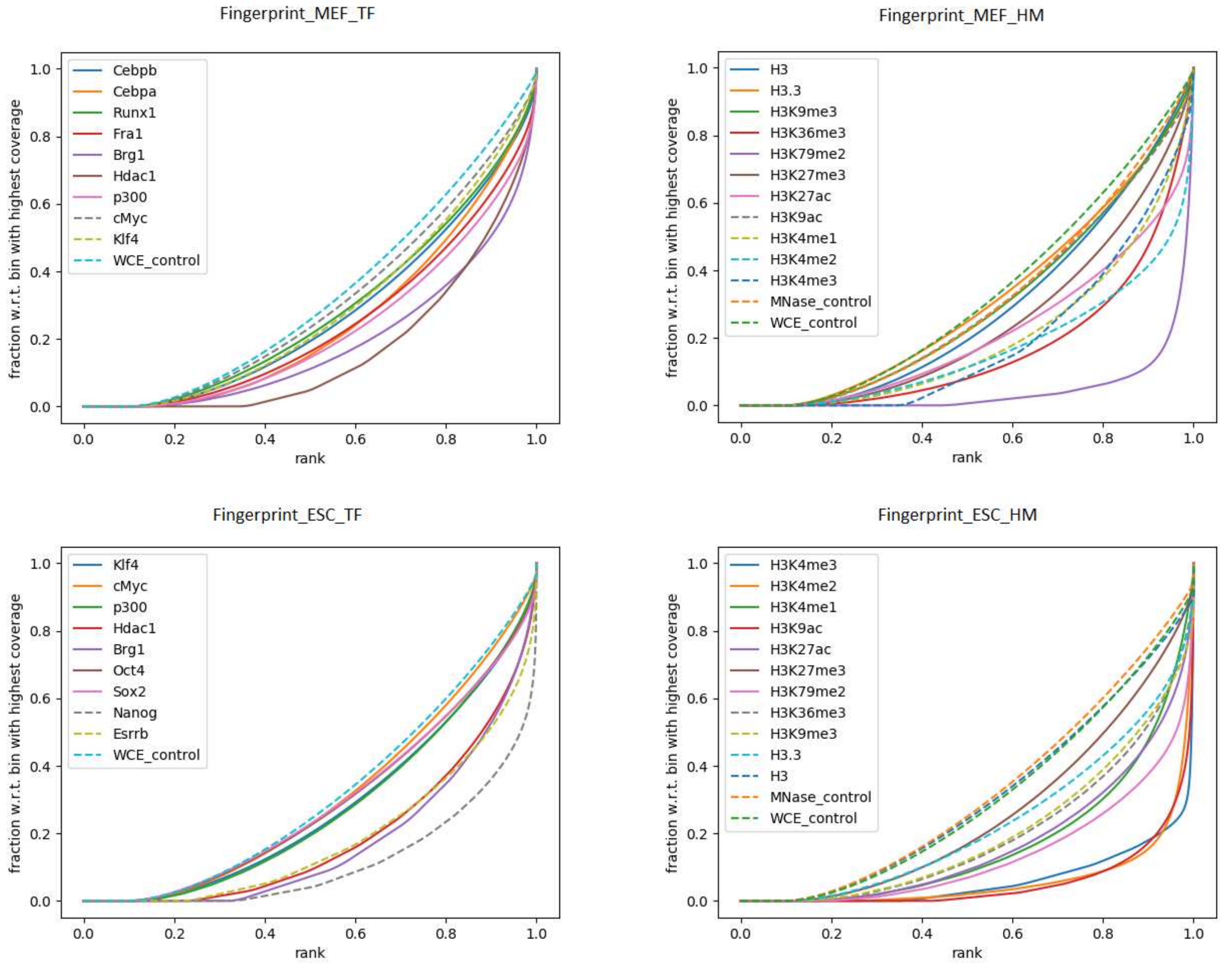


Fig. S2. fingerprints of histone modifications and transcription factors in MEF and ESC

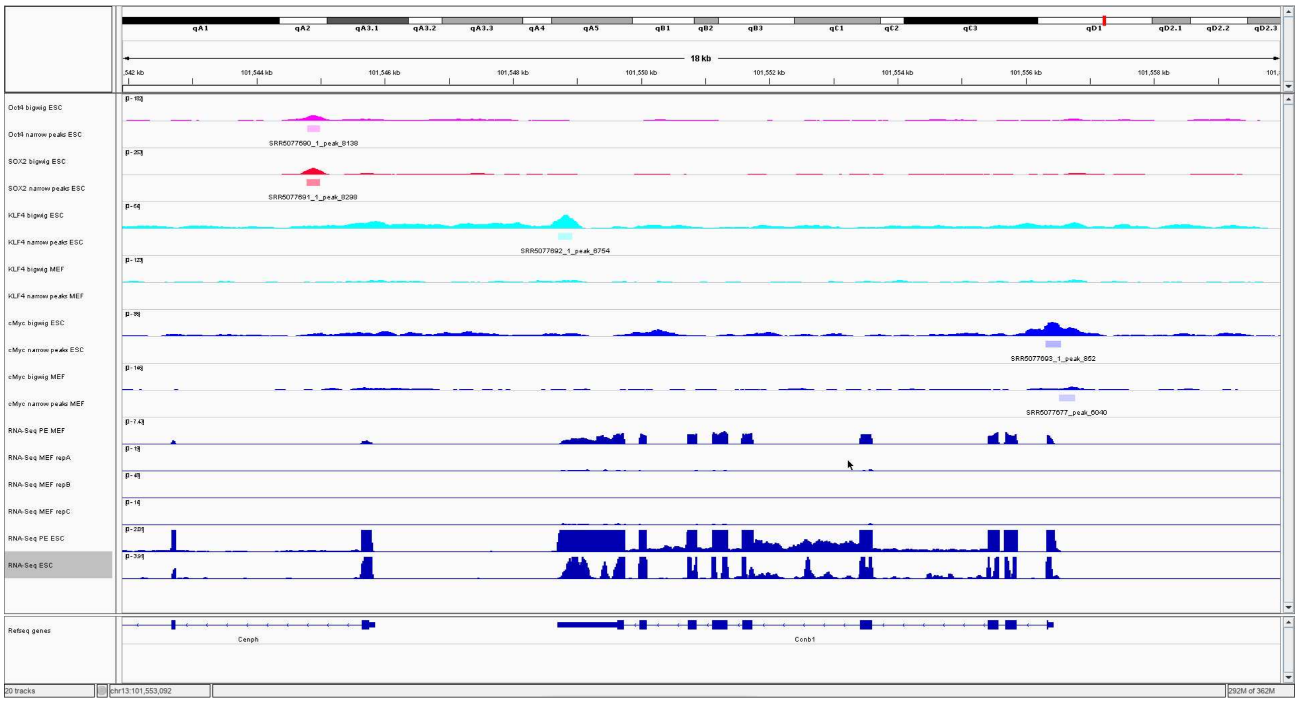


Fig. S3. IGV Image of signals in .bigWig, Rna-seq and peak files

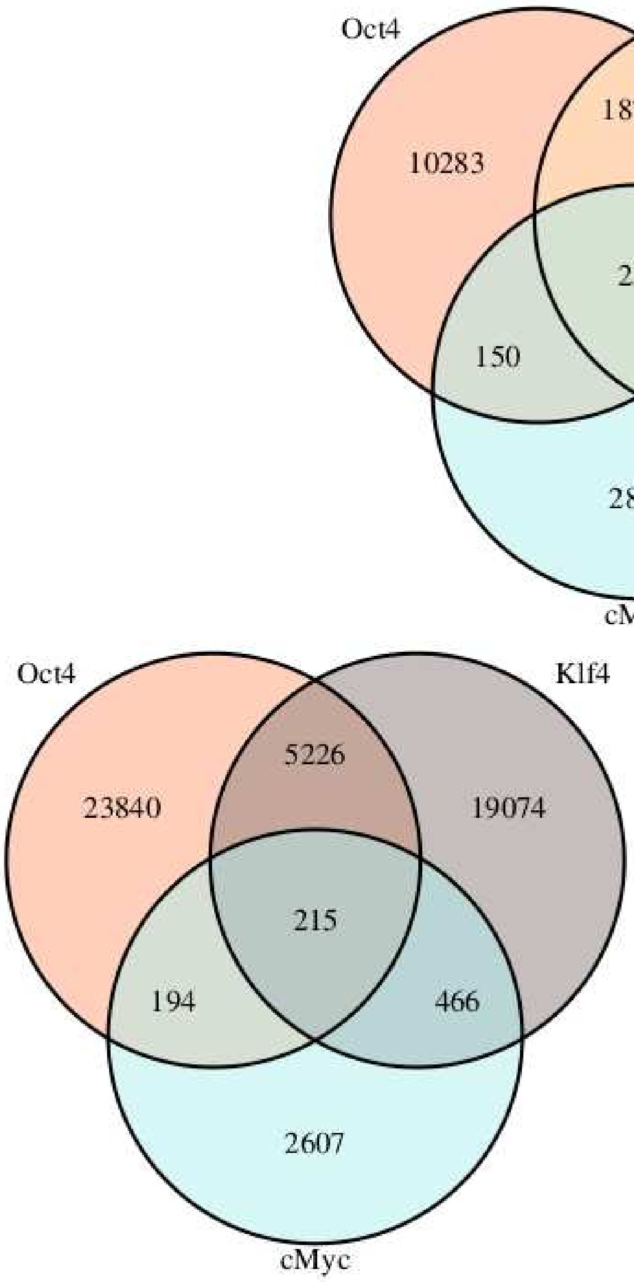


Fig. S4. Shared peaks of different transcription factors in ESC

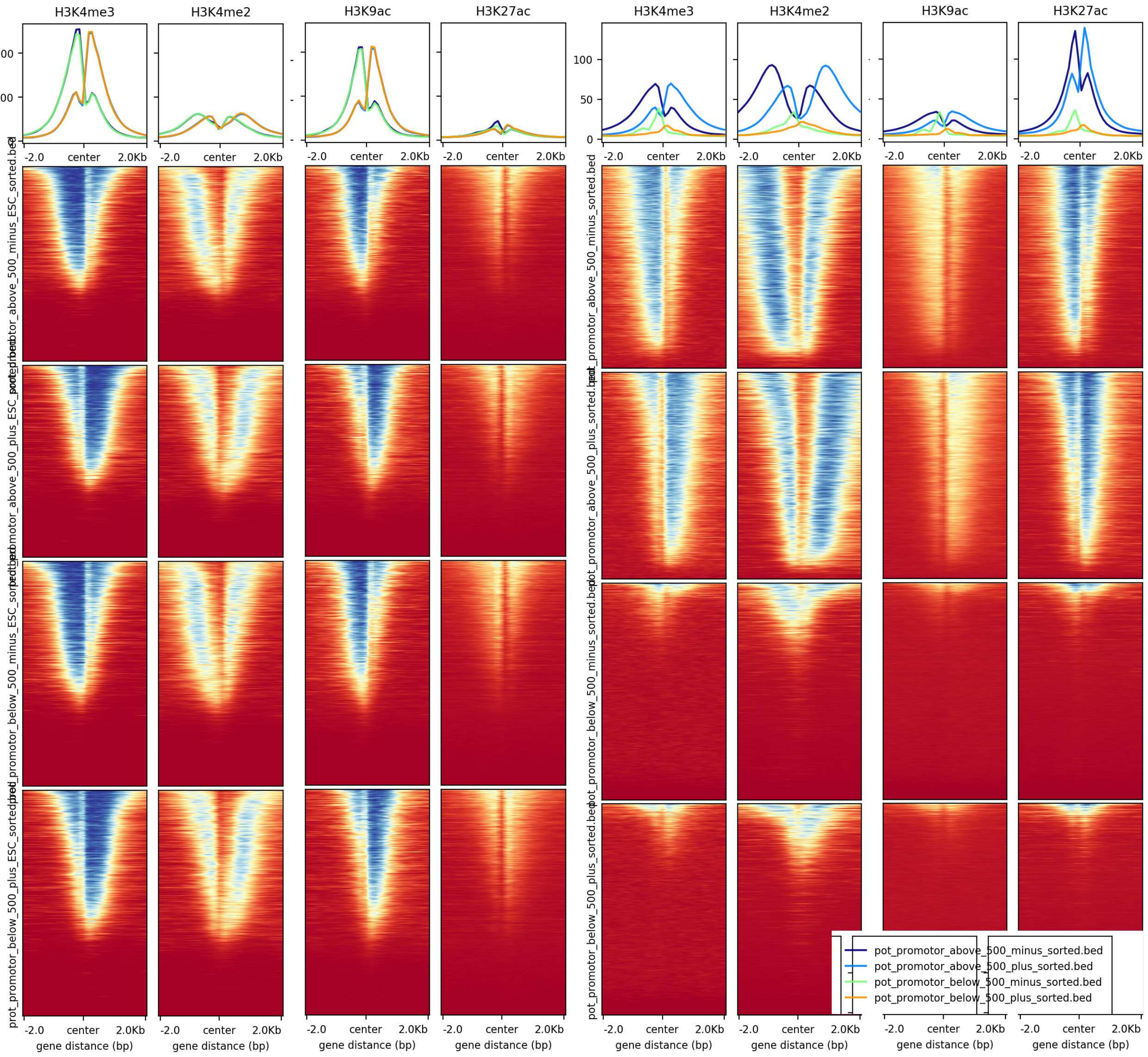


Fig. S5. Heatmap with most remarkable signals in promoter region according to the strands in ESC and MEF

Table S1. Number of significant peaks in all experiments

HM/TF	**MEF**	**ESC**
H3K4me3	23124	42355
H3K4me2	147290	171863
H3K4me1	261130	199966
H3K9ac	30337	122129
H3K27ac	105054	106031
H3K27me3	88768	24295
H3K79me2	91976	67879
H3K36me3	388342	98844
H3K9me3	7037	65494
H3.3	16583	61574
H3	132	0
p300	97458	13787
Hdac1	37438	48029
Klf4	20209	24981
cMyc	22039	3482
Brg1	139174	30365
Fra1	68117	
Runx1	33612	
Cebpa	24036	
Cebpb	48592	
Oct4		29475
Sox2		29785
Nanog		72111
Esrrb		59666
HM	1294669	1022246
TF	355779	249865
total	1650448	1272111

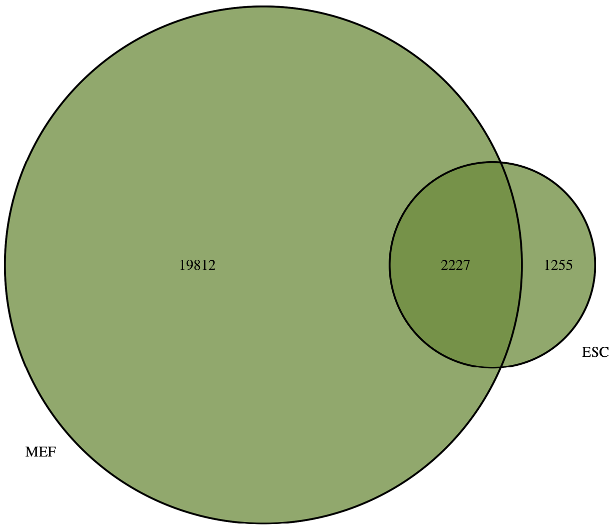


Fig. S6. Number of unique and shared of cMyc peaks in MEF and ESC

Table S2. β coefficients of the multivariate regression for ESC

modification	value	p-value
(Intercept)	0.16989	0.0584
H3K4me3	0.57822	< 2e-16
H3K4me2	-0.26536	< 2e-16
H3K4me1	0.35192	0.< 2e-16
H3K9ac	0.42256	< 2e-16
H3K27ac	0.49311	< 2e-16
H3K27me3	-0.52931	< 2e-16
H3K79me2	0.03307	0.2481
H3K36me3	-0.01427	0.6122
H3K9me3	-0.27798	1.19e-13

```r
### library biomaRt for the genome annotation, we have to choose the corresponding version of the mm9 genome ##

library(biomaRt)

ensembl67 <- useMart(host=’may2012.archive.ensembl.org’, biomart=’ENSEMBL_MART_ENSEMBL’, dataset=’mmusculus_gene_ensembl’)

### now get all (protein coding) genes

prot.gene <- getBM(attributes=c("ensembl_gene_id", "gene_biotype", "strand",  "chromosome_name", "start_position", "end_position"),mart = ensembl67, filters = "biotype", values = "protein_coding")

### now the data frame prot.gene has to be converted into GenomicRanges object. The chromosome names and the strand has to be changed before, as an example, two simple functions....


convert.strand <- function(strand.col){
  xx <-strand.col
  xx <- as.factor(xx)
  levels(xx)[levels(xx)=="-1"] <- "-"
  levels(xx)[levels(xx)=="1"] <- "+"
  levels(xx)[!levels(xx)%in%c("-","+")] <- "*"
  return(xx)
}


correct.chr <- function(genes){
  genes <- genes[genes$chromosome_name%in%c(1:19,"X","Y"),]
  genes$chromosome_name <- paste0("chr",genes$chromosome_name)
  return(genes)
}


library(GenomicRanges)

prot.gene$strand <- convert.strand(prot.gene$strand)
prot.gene <- correct.chr(prot.gene)
prot.gene.gr <- makeGRangesFromDataFrame(df=prot.gene, start.field="start_position", end.field="end_position", keep.extra.columns=TRUE,starts.in.df.are.0based=FALSE) #ensembl:1-based


## get the promoters with promoters function

upstr = 2000
downstr = 500
prot.gene.prom <- promoters(prot.gene.gr, upstream = upstr, downstream = downstr)

setwd("/project/functional-genomics/2019/data/sra/MEF_G3/prefetched/mapped/")
library(bamsignals)
print("get the readcounts from our different histone modifications")
MEF_H3K4me3_counts <- bamCount("SRR5077625_rmDup_sorted.bam", prot.gene.prom, verbose=F)
```

```
MEF_H3K4me2_counts <- bamCount("SRR5077629_rmDup_sorted.bam", prot.gene.prom, verbose=F)
MEF_H3K4me1_counts <- bamCount("SRR5077633_rmDup_sorted.bam", prot.gene.prom, verbose=F)
MEF_H3K9ac_counts <- bamCount("SRR5077637_rmDup_sorted.bam", prot.gene.prom, verbose=F)
MEF_H3K27ac_counts <- bamCount("SRR5077641_rmDup_sorted.bam", prot.gene.prom, verbose=F)
MEF_H3K27me3_counts <- bamCount("SRR5077645_rmDup_sorted.bam", prot.gene.prom, verbose=F)
MEF_H3K79me2_counts <- bamCount("SRR5077649_rmDup_sorted.bam", prot.gene.prom, verbose=F)
MEF_H3K36me3_counts <- bamCount("SRR5077653_rmDup_sorted.bam", prot.gene.prom, verbose=F)
MEF_H3K9me3_counts <- bamCount("SRR5077657_rmDup_sorted.bam", prot.gene.prom, verbose=F)
MEF_MNase_counts <- bamCount("SRR5077669_rmDup_sorted.bam", prot.gene.prom, verbose=F)
MEF_WCE_counts <- bamCount("SRR5077673_rmDup_sorted.bam", prot.gene.prom, verbose=F)

gene_id <- prot.gene.prom$ensembl_gene_id

hm_count <- data.frame(H3K4me3=MEF_H3K4me3_counts,
                       H3K4me2=MEF_H3K4me2_counts,
                       H3K4me1=MEF_H3K4me1_counts,
                       H3K9ac=MEF_H3K9ac_counts,
                       H3K27ac=MEF_H3K27ac_counts,
                       H3K27me3=MEF_H3K27me3_counts,
                       H3K79me2=MEF_H3K79me2_counts,
                       H3K36me3=MEF_H3K36me3_counts,
                       H3K9me3=MEF_H3K9me3_counts,
                       MNase=MEF_MNase_counts,
                       WCE=MEF_WCE_counts)

#get the readcounts from our different histone modifications from ESC
setwd("/project/functional-genomics/2019/data/sra/MEF_G3/prefetched/mapped_ESC/")
ESC_H3K4me3_counts <- bamCount("SRR5077628_1_rmDup_sorted.bam", prot.gene.prom, verbose=F)
ESC_H3K4me2_counts <- bamCount("SRR5077632_1_rmDup_sorted.bam", prot.gene.prom, verbose=F)
ESC_H3K4me1_counts <- bamCount("SRR5077636_1_rmDup_sorted.bam", prot.gene.prom, verbose=F)
ESC_H3K9ac_counts <- bamCount("SRR5077640_1_rmDup_sorted.bam", prot.gene.prom, verbose=F)
ESC_H3K27ac_counts <- bamCount("SRR5077644_1_rmDup_sorted.bam", prot.gene.prom, verbose=F)
ESC_H3K27me3_counts <- bamCount("SRR5077648_1_rmDup_sorted.bam", prot.gene.prom, verbose=F)
ESC_H3K79me2_counts <- bamCount("SRR5077652_1_rmDup_sorted.bam", prot.gene.prom, verbose=F)
ESC_H3K36me3_counts <- bamCount("SRR5077656_1_rmDup_sorted.bam", prot.gene.prom, verbose=F)
ESC_H3K9me3_counts <- bamCount("SRR5077660_1_rmDup_sorted.bam", prot.gene.prom, verbose=F)
ESC_MNase_counts <- bamCount("SRR5077672_1_rmDup_sorted.bam", prot.gene.prom, verbose=F)
ESC_WCE_counts <- bamCount("SRR5077675_1_rmDup_sorted.bam", prot.gene.prom, verbose=F)

ESC_hm_count <- data.frame(H3K4me3=ESC_H3K4me3_counts,
                       H3K4me2=ESC_H3K4me2_counts,
                       H3K4me1=ESC_H3K4me1_counts,
                       H3K9ac=ESC_H3K9ac_counts,
                       H3K27ac=ESC_H3K27ac_counts,
                       H3K27me3=ESC_H3K27me3_counts,
                       H3K79me2=ESC_H3K79me2_counts,
                       H3K36me3=ESC_H3K36me3_counts,
                       H3K9me3=ESC_H3K9me3_counts,
                       MNase=ESC_MNase_counts,
                       WCE=ESC_WCE_counts)

normalize_signal <- function(sample_count, control_count){
  S <- (sample_count + 1)
  C <- (control_count + 1)
  m <- median(S/C)
  Snorm <- S/C * 1/m
  Snorm_log2 <- log(Snorm)
  Snorm_log2_scaled <- scale(Snorm_log2-mean(Snorm_log2) + 1, center=F, scale=T)
  return(list("log"= Snorm_log2, "scaled"= Snorm_log2_scaled))
}

normalize_signal <- function(sample_count, control_count){
  Snorm_log2 <- log(sample_count+1)
  Snorm_log2_scaled <- scale(Snorm_log2-mean(Snorm_log2) + 1, center=F, scale=T)
  return(list("log"= Snorm_log2, "scaled"= Snorm_log2_scaled))
}
#normalize the signal for our different histone modifications
hm_count_norm_log <- data.frame(H3K4me3=normalize_signal(MEF_H3K4me3_counts,MEF_MNase_counts)$log,
                       H3K4me2=normalize_signal(MEF_H3K4me2_counts,MEF_MNase_counts)$log,
                       H3K4me1=normalize_signal(MEF_H3K4me1_counts,MEF_MNase_counts)$log,
                       H3K9ac=normalize_signal(MEF_H3K9ac_counts,MEF_MNase_counts)$log,
                       H3K27ac=normalize_signal(MEF_H3K27ac_counts,MEF_MNase_counts)$log,
                       H3K27me3=normalize_signal(MEF_H3K27me3_counts,MEF_MNase_counts)$log,
                       H3K79me2=normalize_signal(MEF_H3K79me2_counts,MEF_WCE_counts)$log,
                       H3K36me3=normalize_signal(MEF_H3K36me3_counts,MEF_MNase_counts)$log,
                       H3K9me3=normalize_signal(MEF_H3K9me3_counts,MEF_WCE_counts)$log)
rownames(hm_count_norm_log) <- gene_id
hm_count_norm_log_sorted <- hm_count_norm_log[ order(rownames(hm_count_norm_log)),]

hm_count_norm_scaled <- data.frame(H3K4me3=normalize_signal(MEF_H3K4me3_counts,MEF_MNase_counts)$scaled,
                                H3K4me2=normalize_signal(MEF_H3K4me2_counts,MEF_MNase_counts)$scaled,
                                H3K4me1=normalize_signal(MEF_H3K4me1_counts,MEF_MNase_counts)$scaled,
                                H3K9ac=normalize_signal(MEF_H3K9ac_counts,MEF_MNase_counts)$scaled,
                                H3K27ac=normalize_signal(MEF_H3K27ac_counts,MEF_MNase_counts)$scaled,
                                H3K27me3=normalize_signal(MEF_H3K27me3_counts,MEF_MNase_counts)$scaled,
                                H3K79me2=normalize_signal(MEF_H3K79me2_counts,MEF_WCE_counts)$scaled,
                                H3K36me3=normalize_signal(MEF_H3K36me3_counts,MEF_MNase_counts)$scaled,
                                H3K9me3=normalize_signal(MEF_H3K9me3_counts,MEF_WCE_counts)$scaled)
rownames(hm_count_norm_scaled) <- gene_id
hm_count_norm_scaled_sorted <- hm_count_norm_scaled[ order(rownames(hm_count_norm_scaled)),]



ESC_hm_count_norm_log <- data.frame(H3K4me3=normalize_signal(ESC_H3K4me3_counts,ESC_MNase_counts)$log,
                                H3K4me2=normalize_signal(ESC_H3K4me2_counts,ESC_MNase_counts)$log,
                                H3K4me1=normalize_signal(ESC_H3K4me1_counts,ESC_MNase_counts)$log,
                                H3K9ac=normalize_signal(ESC_H3K9ac_counts,ESC_MNase_counts)$log,
                                H3K27ac=normalize_signal(ESC_H3K27ac_counts,ESC_MNase_counts)$log,
                                H3K27me3=normalize_signal(ESC_H3K27me3_counts,ESC_MNase_counts)$log,
                                H3K79me2=normalize_signal(ESC_H3K79me2_counts,ESC_WCE_counts)$log,
                                H3K36me3=normalize_signal(ESC_H3K36me3_counts,ESC_MNase_counts)$log,
                                H3K9me3=normalize_signal(ESC_H3K9me3_counts,ESC_WCE_counts)$log)
rownames(ESC_hm_count_norm_log) <- gene_id
ESC_hm_count_norm_log_sorted <- ESC_hm_count_norm_log[ order(rownames(ESC_hm_count_norm_log)),]

ESC_hm_count_norm_scaled <- data.frame(H3K4me3=normalize_signal(ESC_H3K4me3_counts,ESC_MNase_counts)$scaled,
                                   H3K4me2=normalize_signal(ESC_H3K4me2_counts,ESC_MNase_counts)$scaled,
                                   H3K4me1=normalize_signal(ESC_H3K4me1_counts,ESC_MNase_counts)$scaled,
                                   H3K9ac=normalize_signal(ESC_H3K9ac_counts,ESC_MNase_counts)$scaled,
                                   H3K27ac=normalize_signal(ESC_H3K27ac_counts,ESC_MNase_counts)$scaled,
                                   H3K27me3=normalize_signal(ESC_H3K27me3_counts,ESC_MNase_counts)$scaled,
                                   H3K79me2=normalize_signal(ESC_H3K79me2_counts,ESC_WCE_counts)$scaled,
                                   H3K36me3=normalize_signal(ESC_H3K36me3_counts,ESC_MNase_counts)$scaled,
                                   H3K9me3=normalize_signal(ESC_H3K9me3_counts,ESC_WCE_counts)$scaled)
rownames(ESC_hm_count_norm_scaled) <- gene_id
ESC_hm_count_norm_scaled_sorted <- ESC_hm_count_norm_scaled[ order(rownames(ESC_hm_count_norm_scaled)),]

library(biomaRt)
library(GenomicFeatures)
# get all exons corresponding to all ensembl genes #
mm9.exons <- makeTxDbFromBiomart(biomart="ENSEMBL_MART_ENSEMBL",
                    dataset="mmusculus_gene_ensembl",
                    transcript_ids=NULL,
                    circ_seqs=DEFAULT_CIRC_SEQS,
                    filter=NULL,
                    id_prefix="ensembl_",
                    host="may2012.archive.ensembl.org", #www.ensembl.org",
                    port=80,
                    taxonomyId=NA,
                    miRBaseBuild=NA)
# now get the exons per gene (list of genomic ranges)
exonic <- exonsBy(mm9.exons, by="gene")
# reduce the exons by the union (list of genomic ranges)
```

```r
red.exonic <- reduce(exonic)
# lengts of all genes as a sum of exons
exon.lengths <- sum(width(red.exonic))


setwd("/project/functional-genomics/2019/group3/rna_mapping/tableReads/")
library("DESeq2")
MEF_repA<- read.table("SRR5077600_ReadsPerGene.out.tab", row.names=1, skip=4)
MEF_repB<- read.table("SRR5077601_ReadsPerGene.out.tab", row.names=1, skip=4)
MEF_repC<- read.table("SRR5077602_ReadsPerGene.out.tab", row.names=1, skip=4)
MEF_PE<- read.table("SRR5077621_ReadsPerGene.out.tab", row.names=1, skip=4)
ESC_PE <- read.table("SRR5077624_ReadsPerGene.out.tab", row.names=1, skip=4)
ESC_SE <- read.table("SRR5077609_ReadsPerGene.out.tab", row.names=1, skip=4)

cts <- data.frame(MEF_PE=MEF_PE[,3], MEF_repA=MEF_repA[,3], MEF_repB=MEF_repB[,3], MEF_repC=MEF_repC[,3], ESC_PE=ESC_PE[,3], ESC_SE=ESC_SE[,3])
rownames(cts) <- rownames(MEF_PE)
cts <- cts[ order(rownames(cts)),]

coldata <- matrix(c(rep("MEF",4),rep("ESC",2),"paired-end",rep("single-read",3),"paired-end","single-end" ), nrow=6)
colnames(coldata) <- c("tissue", "type")
rownames(coldata) <- colnames(cts)
print("construct dds")
dds <- DESeqDataSetFromMatrix(countData=cts, colData=DataFrame(coldata), design= ~tissue)
dds <- estimateSizeFactors(dds)
### you can add this information to the dds object of DESeq2 ##
rowRanges(dds) <- red.exonic
## and then just use the function to calculate FPKM values
print("calculate fpkm")
fpkm.dds <- fpkm(dds)
print("filter for protein coding genss")
prot.indices <- rownames(fpkm.dds) %in% prot.gene$ensembl_gene_id
fpkm.prot.dds <- fpkm.dds[prot.indices,]
print("calculate medians")
Y <- as.matrix(rowMedians(fpkm.prot.dds[,1:4]))#uses mean because we have 4 columns
rownames(Y) <- rownames(fpkm.prot.dds)
fpkm.prot.dds_wo_PE <- cbind(fpkm.prot.dds[,2:4])
Y_wo_PE <- as.matrix(rowMedians(fpkm.prot.dds_wo_PE)) #"real" median but without paired end data
rownames(Y_wo_PE) <- rownames(fpkm.prot.dds)
print("use normalized counts instead of fpkm")
norm.counts <- counts(dds,normalized=T)
norm.counts.prot <- norm.counts[prot.indices,]
Y_norm.count <- as.matrix(rowMedians(log(norm.counts.prot[,1:4]+1)))
rownames(Y_norm.count) = rownames(norm.counts.prot)

norm.counts.prot_wo_PE <- norm.counts.prot[,2:4]
Y_norm.count_wo_PE <- as.matrix(rowMedians(log2(norm.counts.prot_wo_PE+1)))
rownames(Y_norm.count_wo_PE) = rownames(norm.counts.prot)

#calculate medians for ESC
ESC_Y <- as.matrix(rowMedians(fpkm.prot.dds[,5:6]))#uses mean because we have 4 columns
rownames(ESC_Y) <- rownames(fpkm.prot.dds)
#use normalized counts instead of fpkm for ESC
ESC_Y_norm.count <- as.matrix(rowMedians(log(norm.counts.prot[,5:6]+1)))
rownames(ESC_Y_norm.count) = rownames(norm.counts.prot)


set.seed(1234)
random_indices <- sample(length(Y_wo_PE)) #random permutation of indices
#constructing training data set
training_indices <- random_indices[1:(length(Y_wo_PE)/2)]
training_X_log <- hm_count_norm_log_sorted[training_indices,]
training_X_scaled <- hm_count_norm_scaled_sorted[training_indices,]

ESC_training_X_log <- ESC_hm_count_norm_log_sorted[training_indices,]
ESC_training_X_scaled <- ESC_hm_count_norm_scaled_sorted[training_indices,]
#constructing validation data set
validation_indices <- random_indices[(length(Y_wo_PE)/2):((length(Y_wo_PE)/2)+(length(Y_wo_PE)/4))]
validation_X_log <- hm_count_norm_log_sorted[validation_indices,]
validation_X_scaled <- hm_count_norm_scaled_sorted[validation_indices,]
```

```

ESC_validation_X_log <- ESC_hm_count_norm_log_sorted[validation_indices,]
ESC_validation_X_scaled <- ESC_hm_count_norm_scaled_sorted[validation_indices,]
#constructing test data set
test_indices <- random_indices[((length(Y_wo_PE)/2)+(length(Y_wo_PE)/4)):length(Y_wo_PE)]
test_X_log <- hm_count_norm_log_sorted[test_indices,]
test_X_scaled <- hm_count_norm_scaled_sorted[test_indices,]

ESC_test_X_log <- ESC_hm_count_norm_log_sorted[test_indices,]
ESC_test_X_scaled <- ESC_hm_count_norm_scaled_sorted[test_indices,]

training_Y <- Y_wo_PE[training_indices]
validation_Y <- Y_wo_PE[validation_indices]
test_Y <- Y_wo_PE[test_indices]

ESC_training_Y <- ESC_Y[training_indices]
ESC_validation_Y <- ESC_Y[validation_indices]
ESC_test_Y <- ESC_Y[test_indices]

training_Y_norm.count <- Y_norm.count_wo_PE[training_indices]
validation_Y_norm.count <- Y_norm.count_wo_PE[validation_indices]
test_Y_norm.count <- Y_norm.count_wo_PE[test_indices]

ESC_training_Y_norm.count <- ESC_Y_norm.count[training_indices]
ESC_validation_Y_norm.count <- ESC_Y_norm.count[validation_indices]
ESC_test_Y_norm.count <- ESC_Y_norm.count[test_indices]


#build linear model for log2 data using fpkm as Y
data_log2 <- as.data.frame(cbind(training_Y,training_X_log))
lm.hist_mod.log2 <- lm(training_Y ~ H3K4me3 + H3K4me2 + H3K4me1 + H3K9ac + H3K27ac + H3K27me3 +
    H3K79me2 + H3K36me3 + H3K9me3 , data = data_log2)
summary(lm.hist_mod.log2)

#build linear model for log2 data with normalized read count as Y
data_log2_norm.count <- as.data.frame(cbind(training_Y_norm.count,training_X_log))
lm.hist_mod.log2_norm.count <- lm(training_Y_norm.count ~ H3K4me3 + H3K4me2 + H3K4me1 + H3K9ac +
    H3K27ac + H3K27me3 + H3K79me2 + H3K36me3 + H3K9me3 , data = data_log2_norm.count)
summary(lm.hist_mod.log2_norm.count)

#build linear model for log2 data with normalized read count as Y for ESC
ESC_data_log2_norm.count <- as.data.frame(cbind(ESC_training_Y_norm.count,ESC_training_X_log))
ESC_lm.hist_mod.log2_norm.count <- lm(ESC_training_Y_norm.count ~ H3K4me3 + H3K4me2 + H3K4me1 + H3K9ac
    + H3K27ac + H3K27me3 + H3K79me2 + H3K36me3 + H3K9me3 , data = ESC_data_log2_norm.count)
summary(ESC_lm.hist_mod.log2_norm.count)

#build linear model for scaled data using fpkm as Y
data_scaled <- as.data.frame(cbind(training_Y,training_X_scaled))
lm.hist_mod.scaled <- lm(training_Y ~ H3K4me3 + H3K4me2 + H3K4me1 + H3K9ac + H3K27ac + H3K27me3 +
    H3K79me2 + H3K36me3 + H3K9me3, data = data_scaled)
summary(lm.hist_mod.scaled)

#build linear model for scaled data with normalized read count as Y
data_scaled_norm.count <- as.data.frame(cbind(training_Y_norm.count,training_X_scaled))
lm.hist_mod.scaled_norm.count <- lm(training_Y_norm.count ~ H3K4me3 + H3K4me2 + H3K4me1 + H3K9ac +
    H3K27ac + H3K27me3 + H3K79me2 + H3K36me3 + H3K9me3 , data = data_scaled_norm.count)
summary(lm.hist_mod.scaled_norm.count)

#build linear model for scaled data with normalized read count as Y for ESC
ESC_data_scaled_norm.count <- as.data.frame(cbind(ESC_training_Y_norm.count,ESC_training_X_scaled))
ESC_lm.hist_mod.scaled_norm.count <- lm(ESC_training_Y_norm.count ~ H3K4me3 + H3K4me2 + H3K4me1 +
    H3K9ac + H3K27ac + H3K27me3 + H3K79me2 + H3K36me3 + H3K9me3 , data = ESC_data_scaled_norm.count)
summary(ESC_lm.hist_mod.scaled_norm.count)


pred_log <- predict.lm(lm.hist_mod.log2_norm.count,newdata = as.data.frame(test_X_log))
names(test_Y_norm.count) <- rownames(test_X_log)
pred_matrix <- cbind(test_Y_norm.count,pred_log)
colnames(pred_matrix) <- c("true","predicted")
cor(pred_matrix)

```

```
ESC_pred_log <- predict.lm(ESC_lm.hist_mod.log2_norm.count,newdata = as.data.frame(ESC_test_X_log))
names(test_Y_norm.count) <- rownames(ESC_test_X_log)
ESC_pred_matrix <- cbind(test_Y_norm.count,ESC_pred_log)
colnames(ESC_pred_matrix) <- c("true","predicted")
cor(ESC_pred_matrix)
```

code for prediction model