# Supplementary methods

## Processing small RNA sequencing data

The small RNA sequencing data were processed with miRDeep2 software (Friedlander et al. 2008). The 3' adapter sequences were trimmed from the sequencing reads. If a read is less than 18 nucleotides (nt) after trimming, it is removed from further analysis. The trimmed reads were mapped onto mouse genome (mm9) using Bowtie software (Langmead et al. 2009), requiring no mismatch in the first 18 nt and no more than 5 alignments in the genome. The best alignment for each mappable read was reported. The miRNA promoters in Figure S7 were identified using a previously described method (Marson et al. 2008).

## Processing of ChIP-seq data

The ChIP-seq reads of 8 epigenomic marks (H3K36me3, H3K27ac, H3K4me1, H3K4me2, H3K4me3, H3K27me3, H2A.Z, and 5-hmC) in three time points were mapped onto the mouse genome (mm9) with Bowtie software (Langmead et al. 2009) allowing 1 mismatch. The number of sequence reads for each genomic segment (200 nt) was counted and then normalized by the total number of mappable reads. The standardized sequence counts were log-transformed, multiplied by 10 and rounded to the nearest integer:

$$v_{w,t,m} = \left[\mathbf{10} \times \boldsymbol{ln}\left(n_{w,t,m} + 1\right)\right],$$

where [n] is the largest integer no larger than n, and $n_{w,t,m}$ is the normalized ChIP-seq read count for epigenomic mark *m* on genomic segment *w* at time *t*.

## Counting ChIP-seq reads in genomic segments

The mouse genome was segregated into 200 nt segments. The number of overlapping sequencing reads on each genomic segment was counted for each ChIP-seq experiment. A genomic segment was judged as not associated an epigenomic mark if the ChIP-seq experiment produced less than 5 reads on this segment.

## Drawing average signals around different groups of regions

The average signals of an epigenomic mark on a list of genomic segments were plotted with the "sitepro" function in CEAS (Cis-regulatory Element Annotation System) (Shin et al. 2009). We set 50 nt as the profiling resolution and 3,000 nt as the size of flanking regions from the center.

## Hierarchical clustering

A hierarchical clustering of GATE identified clusters was done with the fitted λ parameters, using the R function hclust with Euclidean distance and average linkage. We wanted to cut the hierarchical tree to derive at least 10 groups and to cut on long branches. Cutting between 27.3 and 30.8 served this purpose, which gave rise to 14 groups (Figure S3).

## The distribution of temporal correlations between two epigenomic marks

The Pearson correlation coefficient (PCC) between two epigenomic marks was calculated for every genomic segment as follows. Each epigenomic mark was represented as a vector of length 3 (3 time points), and a PCC was calculated for two vectors. Promoter segments were pulled together and an empirical distribution of their PCCs were plotted (red curve, Figure S9). Similarly, empirical distributions for enhancer and gene body segments were derived.

To generate a background distribution of the PCCs, time-independent epigenomic data were simulated with the estimated Poisson parameters of the two epigenomic marks. PCCs were computed from these time-independent data and were used to derive the background distribution.

## The temporal correlations of $^m$CpG, $^m$CpH, and 5-hmC

MeDIP-seq and MRE-seq data were used to estimate to the intensities of $^m$CpG and $^m$CpH. Let *Intensity[$^u$CpG:MRE]* represent MRE-seq read counts and *Intensity[$^m$CpN:MeDIP]* represent MeDIP-seq read counts. We define

*Intensity[$^m$CpG] = Intensity[CpG] - Intensity[$^u$CpG:MRE]*, and

*Intensity[$^m$CpH] = Intensity[$^m$CpN:MeDIP] -* (*Intensity[CpG] - Intensity[$^u$CpG:MRE]*),

where *Intensity[CpG]* is a constant. This constant does not affect the calculation of temporal correlations, and thus

*PCC[5-hmC, $^m$CpG] = PCC[5-hmC, -MRE]*, and

*PCC[5-hmC, $^m$CpH] = PCC[5-hmC, MeDIP+MRE]*,

where *5-hmC*, *MRE*, and *MeDIP* and normalized read counts. *PCC* is the temporal Pearson correlation defined in the previous section.

## The EM algorithm for parameter estimation

### E-step

Let $z_{w,k}$ be the (0,1) cluster membership indicator of genomic segment $w$. $z_{w,k} = 1$ if genomic segment $w$ is in cluster $k$; otherwise, $z_{w,k} = 0$. $z_{w,k}$ is the missing data. The Q-function is

$Q\left(\Lambda, \Lambda^{(old)}\right) = \sum_{w=1}^{W} \sum_{C_w=1}^{K} T_{w,C_w}^{(old)} \left[log\pi_{C_w} + logf(V_w|\lambda, b, C_w)\right],$

where $\Lambda = \{\pi, b, \lambda\}$ and

$$T_{w,k}^{(old)} = P\left(Z_{w,k} = 1 \middle| V_w = v_w, \Lambda^{(old)}\right) = \frac{\pi_k^{(old)} f(V_w|\lambda_k^{(old)}, b_k^{(old)}, k)}{\sum_{s=1}^{K} \pi_s^{(old)} f(V_w|\lambda_s^{(old)}, b_s^{(old)}, s)}$$

$$f(V_w|\lambda_k, b_k, k) = \sum_{H_w}\left[\left(\prod_{t=1}^{T-1} b^k_{H_{w,t},H_{w,t+1}}\right)\left(\prod_{t=1}^{T}\prod_{m=1}^{M}\frac{{\lambda^k_{H_{w,t},m}}^{v_{w,t,m}} e^{-\lambda^k_{H_{w,t},m}}}{v_{w,t,m}!}\right)\right]$$

where $\Lambda^{(old)}$ is the value of $\Lambda$ obtained from the previous step.

**M-step**

The parameter estimate $\Lambda^{(new)}$ is obtained by maximizing the Q-function.

$\Lambda^{(new)} = \left(\pi^{(new)}, \lambda^{(new)}, \mathrm{b}^{(new)}\right) = \mathrm{argmax}(\mathrm{Q}\left(\Lambda, \Lambda^{(\mathrm{old})}\right)).$

In particular, $\pi_k^{(new)} = \frac{\sum_{w=1}^{W} T_{w,k}^{(\mathrm{old})}}{\sum_{w=1}^{W}\sum_{s=1}^{K} T_{w,s}^{(\mathrm{old})}} = \frac{1}{W}\sum_{w=1}^{W} T_{w,k}^{(\mathrm{old})},$

where $z_{w,k}^{(\mathrm{new})} = 1$, if $k = \underset{\mathrm{s}}{\mathrm{argmax}}\left(\boldsymbol{E}\left(z_{w,s}\middle|\boldsymbol{O}, \Lambda\right)\right)$, otherwise $z_{w,k}^{(\mathrm{new})} = 0$. $\boldsymbol{O}$ is the whole dataset. Furthermore, $\left(\lambda_k^{(new)}, b_k^{(new)}\right) = argmax_{\lambda_k, b_k} \sum_{w=1}^{W} z_{w,k}^{(new)} \{logf(V_w|\lambda_k, b_k, k)\}$ (1).

The Baum-Welch algorithm (Miklos and Meyer 2005) is used to maximize (1) and obtain parameter estimate $\hat{\Lambda}$.

## Implementing a Baum-Welch algorithm

In a Baum-Welch algorithm, two probabilities are computed iteratively, namely the forward probability and the backward probability.

**Forward probability**

Let $F_{w,t}^{k}(i) = P\ (v_{w,1}, v_{w,2}, \dots, v_{w,t}, H_{w,t} = i|C_w = k, \Lambda)$. We have

$$F_{w,t+1}^{k}(j) = P\left(v_{w,t+1}\middle|H_{w,t+1} = j, C_w = k, \lambda\right) \times \sum_{i=0}^{1} F_{w,t}^{k}(i) \times b_{i,j}^{k} = \left(\prod_{m=1}^{M} \frac{\left(\lambda_{j,m}^{k}\right)^{v_{w,t+1,m}} e^{-\lambda_{j,m}^{k}}}{v_{w,t+1,m}!}\right) \sum_{i=0}^{1} F_{w,t}^{k}(i) \times b_{i,j}^{k}\ .$$

Because $H_{w,1} = 0$ for all $w$, we have

$$F_{w,1}^{k}(0) = \ P\left(\nu_{w,1} \middle| H_{w,1} = 0, C_w = k, \lambda\right), \text{and } F_{w,1}^{k}(1) = 0.$$

Finally,

$$P\ (\nu_w | C_w = k, \Lambda) = \ F_{w,3}^{k}(1) + \ F_{w,3}^{k}(0),$$

where $\nu_w = (\nu_{w,1}, \nu_{w,2}, \nu_{w,3})$, representing the data from 3 time points.

**Backward probability**

Let $B_{w,t}^{k}(i) = P\ \left(\nu_{w,t+1}, \nu_{w,t+2}, \dots, \nu_{w,3} \middle| H_{w,t} = i, C_w = k, \Lambda\right)$. $B_{w,t}^{k}(i)$ can be computed as

$$B_{w,t}^{k}(i) = \ \sum_{j=0}^{1} B_{w,t+1}^{k}(j) \times b_{i,j}^{k} \times P\left(\nu_{w,t+1} \middle| H_{w,t+1} = j, C_w = k, \lambda\right)$$

$$= \ \textstyle\sum_{j=0}^{1} B_{w,t+1}^{k}(j) \times b_{i,j}^{k} \ \times \left( \prod_{m=1}^{M} \frac{\left(\lambda_{j,m}^{k}\right)^{\nu_{w,t+1,m}} e^{-\lambda_{j,m}^{k}}}{\nu_{w,t+1,m}!} \right),$$

where $i, j \in (0,1)$. In particular, $B_{w,3}^{k}(0) = \ B_{w,3}^{k}(1) = 1$.

## Computational details

**Parameter re-estimation: a Baum-Welch algorithm for estimating b, $\boldsymbol{\lambda}$ and H.**

Within an EM step (Step $l$), we call the maximization step as Step $s_l$. In this step, a Baum-Welch algorithm is implemented to maximize (1) and estimate b, λ and H. After calculating the forward and backward probabilities for the HMM in each time point, the joint conditional probability for each genomic segment is specified as:

$$\xi^{k}{}_{w,t}{}^{(\,s_l)}(i,j) = p\left\{H_{w,t} = i, H_{w,t+1} = j \,\middle|\nu_w, C_w = k, \Lambda^{(\,(s-1)_l)}\right\} = \frac{p\left\{H_{w,t}=i, H_{w,t+1}=j, \nu_w \,\middle|C_w=k, \Lambda^{(\,(s-1)_l)}\right\}}{\sum_{i=0}^{1}\sum_{j=0}^{1} p\left\{H_{w,t}=i, H_{w,t+1}=j, \nu_w \,\middle|C_w=k, \Lambda^{(\,(s-1)_l)}\right\}}, \text{ where } i, j = 0, 1.$$

The marginal conditional probability for $H_{w,t}$ is

$$\gamma^{k}{}_{w,t}{}^{(\,s_l)}(i) = p\left\{H_{w,t} = i \,\middle|\nu_w, C_w = k, \Lambda\right\} = \frac{p\left\{H_{w,t} = i, \nu_w \,\middle|C_w = k, \Lambda^{(\,(s-1)_l)}\right\}}{\sum_{i=0}^{1} p\left\{H_{w,t} = i, \nu_w \,\middle|C_w = k, \Lambda^{(\,(s-1)_l)}\right\}},$$

where $i = 0, 1$.

$b$ is estimated by

$$b_{i,j}^{k}{}^{(s_l)} = \frac{\sum_{w=1}^{W}\sum_{t=1}^{2}\left({\xi^k}_{w,t}{}^{(s_l)}(i,j) \times {z_{w,k}}^{(l-1)}\right)}{\sum_{w=1}^{W}\sum_{t=1}^{2}\left({\gamma^k}_{w,t}{}^{(s_l)}(i) \times {z_{w,k}}^{(l-1)}\right)},$$

where ${z_{w,k}}^{(l-1)}$ is the z value got from Step $l-1$.

With marginal posterior probability ${\gamma^k}_{w,t}{}^{(s_l)}(i)$, each ${H_{w,t}}^{(s_l)}$ is estimated by:

$${H_{w,t}}^{(s_l)} = argmax_i\left[{\gamma^k}_{w,t}{}^{(s_l)}(i)\right],$$

and

$$\lambda_{i,m}^{k}{}^{(s_l)} = \frac{\sum_{w=1}^{W}\sum_{t=1}^{3}\left(\left|i+{H_{w,t}}^{(s_l)}-1\right| \times \nu_{w,t,m} \times {z_{w,k}}^{(l-1)}\right)}{\sum_{w=1}^{W}\sum_{t=1}^{3}\left(\left|i+{H_{w,t}}^{(s_l)}-1\right| \times {z_{w,k}}^{(l-1)}\right)}.$$

**Re-clustering genomic segments and estimating z, π**

After estimating $b^{(l)}$ and $\lambda^{(l)}$ in Step $l$, $z^{(l)}$ and $\pi^{(l)}$ can be updated as

$$\pi_k^{(l)} = \frac{1}{W}\sum_{w=1}^{W}\boldsymbol{E}\left(z_{w,k}\middle|\boldsymbol{O},\Lambda\right) = \frac{1}{W}\sum_{w=1}^{W}\frac{\left(F_{w,3}^{k}{}^{(l)}(1) + F_{w,3}^{k}{}^{(l)}\right)\times\pi_k^{(l-1)}}{\sum_{s=1}^{K}\left(F_{w,3}^{s}{}^{(l)} + F_{w,3}^{s}{}^{(l)}\right)\times\pi_s^{(l-1)}}.$$

${z_{w,k}}^{(l)}$ is updated using $\pi_k^{(l)}$ by

$${z_{w,k}}^{(l)} = \begin{cases} 1 & if\ k = \underset{\mathrm{l}}{\operatorname{argmax}}\,\boldsymbol{E}\left(z_{w,l}\middle|\boldsymbol{O},\Lambda\right) = \operatorname{argmax}_l\left(P(\nu_w, C_w = l|\Lambda)\right) \\ 0 & otherwise \end{cases}.$$

## Simulation study

We simulated 4 epigenomic marks on 6,000 genomic segments from 4 clusters. Epigenomic data were simulated in three time points. Each data point in cluster $k$ was sampled from a HMM with the transition probability matrix $b^k$ and the emission distribution of Poisson($\lambda_{i,m}^k$), where *i* is the hidden state and *m* is the epigenomic mark (Table S2).

The simulated data mimicked real data in three aspects. First, the real data formed 4 large clusters (groups) corresponding to enhancers, promoters, gene bodies, and repeats. Second, the simulated data had different temporal patterns for different clusters. Furthermore, different hidden states of the same genomic segment emitted data with different emitting distributions, mimicking the change of regulatory functions. Finally, the number of simulated time points is the same as the time points of the ES cell differentiation experiment.

The simulated data were provided to the GATE model as input data. The parameters used for the program are: maxiteration = 1000, nstep = 20, ndistance = 0.001, initial = 2. We ran the program for cluster numbers 2 to 8 and used BIC to choose the best cluster number. Four was found to be the best cluster number.

We compared the clustering accuracy of GATE with K-means algorithm. The average misclassification rate (the proportion of genomic segments that are incorrectly clustered) of K-means was 23.91%, which was 133 times larger than that of GATE (0.18%). The optimized cluster number (4) was used for k-means clustering with the algorithm of Hartigan and Wong.

The GATE estimated parameters $(\lambda, b)$ were close to real parameters (Table S3). More importantly, 99.10% of the hidden states were correctly predicted (Table S4). This is a useful feature because the hidden states reflect when the regulatory function changes.

## Reference

Friedlander MR, Chen W, Adamidi C, Maaskola J, Einspanier R, Knespel S, Rajewsky N. 2008. Discovering microRNAs from deep sequencing data using miRDeep. *Nat Biotechnol* **26**(4): 407-415.

Langmead B, Trapnell C, Pop M, Salzberg SL. 2009. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol* **10**(3): R25.

Marson A, Levine SS, Cole MF, Frampton GM, Brambrink T, Johnstone S, Guenther MG, Johnston WK, Wernig M, Newman J et al. 2008. Connecting microRNA genes to the core transcriptional regulatory circuitry of embryonic stem cells. *Cell* **134**(3): 521-533.

Miklos I, Meyer IM. 2005. A linear memory algorithm for Baum-Welch training. *BMC bioinformatics* **6**: 231.

Shin H, Liu T, Manrai AK, Liu XS. 2009. CEAS: cis-regulatory element annotation system. *Bioinformatics* **25**(19): 2605-2606.