Motivation: We have witnessed an enormous increase in ChIP-Seq data for histone modifications in the past few years. Discovering significant patterns in these data is an important problem for understanding biological mechanisms.

Results: We propose probabilistic partitioning methods to discover significant patterns in ChIP-Seq data. Our methods take into account signal magnitude, shape, strand orientation and shifts. We compare our methods with some current methods and demonstrate significant improvements, especially with sparse data. Besides pattern discovery and classification, probabilistic partitioning can serve other purposes in ChIP-Seq data analysis. Specifically, we exemplify its merits in the context of peak finding and partitioning of nucleosome positioning patterns in human promoters.

Availability and implementation: The software and code are available in the supplementary material.

Contact: Philipp.Bucher@sib-sib.ch

Supplementary information: Supplementary data are available at Bioinformatics online.

Received on July 3, 2013; revised on March 28, 2014; accepted on April 30, 2014

1 INTRODUCTION

ChIP-Seq (immunoprecipitation combined with high-throughput DNA sequencing) experiments allow to characterize in vivo transcription-factor binding events and local chromatin organization on a genome-wide scale (Barski et al., 2007; Johnson et al., 2007; Mardis, 2007). Within the past few years, ChIP-Seq has become a widely used and indispensable technology in the study of transcriptional regulation. Other epigenetic profiling assays are also starting to have a similar impact on the research field (Ku et al., 2011).

A ChIP-Seq experiment produces a large number of sequence tags that are mapped to the genome, resulting in a genome-wide profile of tag counts. A high tag count at a location on the chromosome indicates the presence of a particular protein at that location. This protein may be a sequence-specific transcription factor, a post-translationally modified histone or some other chromatin-associated protein. The regions enriched in ChIP-Seq tags are diverse in terms of magnitude, shape and orientation (Landt et al., 2012). Sequence-specific transcription factors typically produce uniform narrow Gaussian peaks, while regions enriched in histone modifications tend to show complex multimodal signal distributions.

The term ‘chromatin signature’ has been coined to designate recurrent patterns found in ChIP-Seq-based histone modification maps and other types of chromatin profiling data (Hon et al., 2009). A chromatin signature is usually represented by a vector of average tag counts in bins of certain sizes (typically 50–500 bp) in a collection of larger genomic regions of sizes 1–10 kb. Chromatin signatures can be detected by so-called aggregation plots (APs) (Jee et al., 2011), if precisely mapped experimentally defined anchor points [e.g. transcription start sites (TSSs)] are available for selection and delineation of the genomic regions of interest. A basic assumption in ChIP-Seq data analysis is that specific chromatin signatures are associated with specific functions. For instance, human promoters are characterized by a nucleosome-free region of ~150 bp and a rigidly positioned H3K4me3-marked +1 nucleosome centered 120 bp downstream from the TSS (Schmid and Bucher, 2007).

Discovering a chromatin signature is difficult, especially when anchor points are not available. An effective algorithm must be capable to cope with the following obstacles.

- **Biological inhomogeneity of the samples:** The set of analyzed genomic regions often consists of multiple unknown subclasses, in which case, a plot derived from all samples shows the superposition of several different chromatin signatures.
- **Alignment uncertainty:** Precise anchor points are rarely available for delineating genomic regions. Selected chromatin regions first need to be optimally shifted (registered) with respect to each other before an AP can reveal a high-resolution chromatin signature.
- **Asymmetry:** Chromatin signatures associated with directional molecular mechanisms (such as transcription) are usually asymmetrical. However, the orientation of the genomic regions is often unknown. The input count vectors should then be compared with each other in both orientations.
- **Sparse count data:** Certain bins may have very low tag counts, leading to high sampling errors.

The problem of inhomogeneity can be tackled by off-the-shelf clustering and partitioning algorithms. In fact, hierarchical clustering and K-means have been incorporated in several
Probabilistic partitioning methods for ChIP-Seq data

multipurpose computational platforms for ChIP-Seq data analysis. SeqMINER (Ye et al., 2011) offers an in-built K-means function, while ChIPseeker (Giannopoulou and Elemento, 2011) is interfaced with a third-party hierarchical clustering software. However, shifting and flipping are only implemented in specialized programs like ChromaSig (Hon et al., 2008), ArchAlign (Lai et al., 2010), CATCHprofiles (Nielsen et al., 2012) and CAGT (Kundaje et al., 2012). ArchAlign performs only shifting and flipping and can find only one single signature. CAGT supports flipping but not shifting. (The problem of optimal shifting is typically solved by exhaustive comparison of all overlapping subregions of a given size from two genomic regions, possibly in both orientations.) ChromaSig, ArchAlign and CATCHprofiles use progressive multiple alignment strategies to assemble similar tag profiles. Because these algorithms have to carry out a large number of pairwise comparisons, they tend to be slow. To overcome this drawback, CAGT applies a two-step divide-and-conquer approach. It first uses the K-median algorithm (a variant of K-means) to define top-level classes and then runs a hierarchical clustering algorithm on each of these classes in turn. The shifting and clustering functions require some type of distance measure. All of these programs, except ChromaSig, use non-probabilistic measures such as the Euclidean distance or the Pearson correlation coefficient, neither of which does well with low counts per measure such as the Euclidean distance or the Pearson correlation coefficient. The underlying principle of this general method is to optimize a mixture model by an Expectation-Maximization (EM) algorithm, a strategy that has already proved effective in finding recurrent DNA motifs in selected genomic regions (Machanick and Bailey, 2011). A key difference in this method compared with the other clustering methods mentioned is that samples are not deterministically assigned to a single class; rather, their classification status is defined by a vector of class membership probabilities. While EM has long been a standard tool in machine learning, it is a general-purpose method, whose convergence rates and running times depend on the exact formulation of the objective function and the updating formulae. The purpose of this article is to demonstrate the merits of EM when applied to ChIP-Seq data and to explain by examples how it can be applied to classification and motif-discovery problems in research on chromatin structures. The probabilistic partitioning approach offers the following advantages.

(1) The use of probabilistic distance functions naturally takes into account random sampling variation in low-count data.
(2) Probabilistic class assignment allows accurate characterization of classes even in situations where the classification of individual samples is uncertain.
(3) Probabilistic class assignment is flexible and can combine goals, for instance, the ranking and prioritizing of ChIP-Seq signal-enriched regions based on peak shape.

(4) Shifting and flipping can be implemented in the EM framework via hidden variables.
(5) The implementation of probabilistic partitioning is straightforward with existing programming platforms. All algorithms used in this work can be implemented by <30 lines of R code.
(6) Flexibility: Methods are readily customized to meet the needs of a particular application. For instance, the switching from a Poisson probabilistic model to a negative binomial model requires only one change in the corresponding R code.
(7) Efficiency: In contrast to most existing methods, the EM algorithm does not require exhaustive pairwise comparisons, so that each iteration runs in time linear in the number of samples.
(8) Transparency and Reproducibility: Methods can be accurately described in a research paper by reproducing a few lines of R code (for example, see the R code given in the Supplementary Material).

Section 2 presents in detail several variants of the probabilistic partitioning algorithms. Section 3 analyses the performance of these algorithms on carefully chosen examples based on simulated and real ChIP-Seq data and compares its performance with K-means clustering and CAGT.

2 METHODS

We are given N samples, S1, S2, ..., SN. These samples could be regions around TSSs of genes or transcription factor binding sites. We divide the genome into bins and count the number of ChIP-Seq fragments that fall into each bin to obtain bin counts. Thus, each sample Si is an integer vector of length L, Si = (si1, si2, ..., sai), where each element sij is a bin count. Bincount vectors of several ChIP-Seq libraries (e.g. different histone marks) may be concatenated to partition them together. We assume that the samples originate from a mixture of K different classes, C1, C2, ..., CK. Each class Cj occurs with characteristic probability pj = P(Cj) and is further characterized by ‘profiles’ of expected bin counts: Cj = (cj1, cj2, ..., cjL).

2.1 EM algorithm

The probability of sample Si given class Cj is computed as follows:

\[ P(S_i | C_j) = \prod_{r=1}^{L} \text{Poisson}(s_{ir}, \lambda = c_{jr}) \]  
(1)

Now, the probability of class Cj given sample Si is given by:

\[ P(C_j | S_i) = \frac{p_j P(S_i | C_j)}{\sum_{k=1}^{K} p_k P(S_i | C_k)} \]  
(2)

Using this probability, we update the classes as follows:

\[ c_{jr} = \frac{\sum_{a=1}^{N} P(C_j | S_a) s_{ar}}{\sum_{a=1}^{N} P(C_j | S_a)} \]  
(3)

\[ p_j = \frac{\sum_{a=1}^{N} P(C_j | S_a)}{N} \]  
(4)
These computations are iteratively carried out for a fixed number of steps.

### 2.2 Modified ‘Shape-Only’ EM algorithm

We also propose a shape-only version of the EM algorithm for normalization purposes. For all \( K \) classes, the average count frequency is set to 1. In other words, we impose

\[
E(C_j) = 1 \iff \sum_{i=1}^{L} c_{ji} = L
\]

Equation (1) is modified as follows:

\[
P(S_i|C_j) = \prod_{i=1}^{L} \text{Poisson} \left( s_{ij}, \lambda_j = c_{ij}/L \sum_{i=1}^{L} s_{ij} \right)
\]

The purpose is to adjust the average count frequency of class \( j \) to the average count value of sample \( i \).

\[
E(\lambda_j) = E(S_i)
\]

We further have to make sure that the average count frequency of the reestimated class \( j \) equals 1. To this end, Equation (3) is modified as follows:

\[
c_j = \frac{\sum_{i=1}^{L} P(C_j|S_i) s_{i}}{\sum_{i=1}^{L} P(C_j|S_i)}
\]

### 2.3 Variations—with shift and flip

We propose some variations of the basic method. In the following, we show how shifting and flipping can be implemented. Note that these two options could be implemented separately. Here (for the sake of generality) we show the version that supports both. Shifting and flipping are modeled with two hidden variables, the shift index \( m \) and the flip state \( inv \).

Let \( m \) be the shift index and \( M \) be the maximum number of shifts allowed, and let \( inv \) be equal to 1 when there is no flip and equal to 2 when there is one. Note that with shifting, the patterns \( C_j \) are shorter than the samples \( S_i \) by \( M-1 \). The notation \( s_{ij}(m, inv) \) will be used to represent the data for a particular shift and flip state: for \( inv = 1 \), \( s_{ij}(m, inv) = S_{i-M}, \ldots, S_{j-1} \); for \( inv = 2 \), \( s_{ij}(m, inv) = S_{i-M+1}, \ldots, S_{j+1} \). Now, the probability of sample \( S_i \) given class \( C_j \) and further conditioned on shift index \( m \) and flip state \( inv \) is computed as follows:

\[
P(S_i|C_j; m, inv) = \prod_{i=1}^{L} \text{Poisson}(s_{ij}(m, inv), \lambda = c_{ij})
\]

Now, the probability of class \( C_j \) given sample \( S_i \) is given by

\[
P(C_j, m, inv|S_i) = \frac{p_j(m, inv)P(S_i|C_j; m, inv)}{\sum_{b=1}^{M} \sum_{d=1}^{2} p_b(d, e)P(S_i|C_b;d,e)}
\]

Using this probability, we update the classes as follows:

\[
c_j = \frac{\sum_{a=1}^{M} \sum_{d=1}^{2} P(C_j; d, e)s_{i}(d, e)}{\sum_{a=1}^{M} \sum_{d=1}^{2} P(C_j; d, e)}
\]

\[
p_j(m, inv) = \frac{\sum_{a=1}^{M} P(C_j, m, inv|S_i)}{N}
\]

Here we assume that the shift states follow a centered Gaussian distribution with equal width for all classes. Therefore, we infer only the SD of the distribution from the data. Practically, this is achieved by applying the following regularization step to the reestimated probabilities \( p_j^*(m, inv) \).

\[
\mu = \frac{\sum_{b=1}^{K} \sum_{d=1}^{M} \sum_{e=1}^{2} p_j^*(d, e)d}{\sum_{b=1}^{K} \sum_{d=1}^{M} \sum_{e=1}^{2} p_j^*(d, e)}
\]

\[
\sigma = \sqrt{\frac{\sum_{b=1}^{K} \sum_{d=1}^{M} \sum_{e=1}^{2} p_j^*(d, e)(d-\mu)^2}{\sum_{b=1}^{K} \sum_{d=1}^{M} \sum_{e=1}^{2} p_j^*(d, e)}}
\]

Let \( \text{Normal}(m|\mu)\) be the probability of class \( m \) that has a Gaussian distribution of mean \( \mu \) and SD \( \sigma \).

\[
p_j(m, inv) = \frac{\text{Normal}(m|\mu)|\sigma}{} \sum_{b=1}^{M} \sum_{d=1}^{2} p_j^*(b, inv)
\]

As before, these computations are iterated for a fixed number of steps.

Because we are able to estimate the probability of each shift for every sample and class, we can use these probabilities to estimate the internal position of a given pattern in a particular sample. Under Section 3.2.3, we present a biological example where we make use of this possibility.

### 2.4 Seeding and initialization strategies

Various seeding and initialization strategies are possible for the proposed probabilistic partitioning algorithms. Here are two such possibilities.

- **Start with one class (K = 1).** Set \( P(C_j|S_i) = 1 \) (for partitioning without shifts or flips) and \( p_1 = 1 \). The initial distribution of class one \( (c_1) \) can be defined in either of the two ways: (i) we can take the mean of the entire data across all the samples; (ii) choose a random distribution by either picking a random subset of the data or by choosing a random probability for each sample, and then taking the weighted sum over all the samples according to their probability value. Then, iteratively increase the number of classes \((K = K + 1)\) till the maximum number of classes is reached. With each iteration, the new class is initialized to a uniform distribution \((c_j = 1) \forall j \) in the new class. The new class will have a prior probability \( \text{P} \text{Prior} = n/K \), where \( K \) is the total number of classes so far. The remaining classes have a total probability \((\sum_{j=1}^{K} p_j) = (1 - 1/K)\), where each class is \( p_j = (1 - 1/K)p_1 \) (the earlier value of \( p_1 \) is \( p_1 \)). After the initialization (for each increase in the number of classes), the EM method is applied.

- **Start with K classes (K ≥ 1).** Like done before, we could take \( K \) different subsets of the original data and compute their mean, and use this to compute the initial distributions for different classes. Alternatively, one could also choose \( K \) random probability vectors (each vector containing probabilities for all samples) and use this to compute \( K \) weighted sums for finding the initial distributions of the \( K \) classes. After this initialization, the EM method is applied.

Determining the optimum number of classes or clusters (choosing \( K \)) in a dataset has been a problem, which has been addressed in the literature for many decades now. The number of classes should strike a balance between assigning all samples into one class and assigning each sample into a separate class. Methods that look at percentage of variance as a function of number of classes (Ketchen and Shook, 1996) or by using methods based on information criteria like Akaike information criterion or Bayesian information criterion are often used (among many others). However, most of these methods have their drawbacks (Yang, 2005).
Because probabilistic partitioning method is to be used as an exploratory tool, we leave it to the user to manually see what is the best number of interesting classes for the dataset being used.

3 RESULTS/DISCUSSION

3.1 On simulated data

We first run the computational experiments on simulated data. The data are composed of a mixture of two classes characterized by bin count frequency profiles of different shapes. The samples were integer vectors of length 100. Counts were generated by randomly sampling from a Poisson distribution with $\lambda$ varying in a class- and position-specific manner along the bin count frequency profiles. Because we were particularly interested in the algorithm’s capability of recovering patterns from sparse count data, we varied the total count coverage $f$ over a wide range of relevant values ($f$ is defined as the total expected bin counts per sample). The simulated data were generated using statistical software R. The R code and additional details of the computational protocols are given in the Supplementary Material.

3.1.1 Data without shifts or flips We first generated random samples belonging to two classes, 1000 samples for each class. The classes were defined by bin count profiles of Gaussian shape, each one with a different mean and variance. The experiments were repeated several times with coverage $f$ ranging from 50 to 0.5. The shape-based version of probabilistic partitioning (Partition) was compared with K-means and the recently introduced Clustered AGgregation Tool (CAGT). The latter was used with two different distance metrics, Euclidean and correlation [henceforth denoted as CAGT (Euclidean) and CAGT (correlation)]. CAGT differs from the other two methods in that it tries to infer the number of classes from the data, a behavior that can be partly controlled by the command line parameter ‘K-means/median’. For the sake of fair comparison, we changed the value of this parameter, so as to force the program to always return exactly two classes. For CAGT (Euclidean), the parameter $k$ (the number of clusters for K-means/medians) was always set to 2, while for CAGT (correlation), it was set to 2 when $f<5$ and to the default value of 40 when $f \geq 5$. For the same reason, we disabled the flipping option with CAGT. During the test, we observed that CAGT (correlation) returned an error when trying to process samples consisting of zeros only. We therefore eliminated these samples from the input datasets fed to CAGT (correlation). The number of EM iterations in the probabilistic partitioning method was set to 30 for any value of $f$. Here and in all subsequent experiments, we used the iterative version of EM, starting with an initial class consisting of the mean bin count vector taken over all samples.

The performance of the different methods was assessed in several ways: (i) by visual inspection of APs for the true and rediscovered classes (Fig. 1)—in the case of the probabilistic partitioning method, the AP represents a probability-weighted average; (ii) by measuring the similarity between the true and rediscovered patterns as a Pearson correlation coefficient of the corresponding bin count profiles (Table 1); (iii) by comparing the reestimated class frequencies to the true class frequencies of 50% (Table 1); (iv) by computing the classification error defined as the percentage of misclassified samples (Table 2). Classification error is calculated as $\frac{(N_{c1} - cr_{c1}) + (N_{c2} - cr_{c2})}{100}$, where $cr_{c1}$ and $cr_{c2}$ are number of samples from classes 1 and class 2, respectively, which were correctly classified as belonging to their respective classes, and $N$ is the total number of samples in the data. To compute the classification error, we need to label the classes inferred by the various algorithms. Because the setup of the simulations involves only two classes, we could easily do this by hand. In addition for the probabilistic partitioning method, we need to give a deterministic class assignment for each sample, and we give it to the most probable class.

As a general trend, we can see that all methods work well when the count coverage is high ($f \geq 10$). When there is a lower coverage, probabilistic partitioning clearly outperforms all other methods. In fact, it recovers the underlying patterns of the two classes surprisingly well ($r>0.94$) even at a low coverage ($f=0.5$) and this in spite of a high classification error of approximately 33% (Table 2). The high classification error is probably due to the expected large number of samples consisting of zeros only (60%) all of which will be attributed to class c2, which has the higher estimated frequency (Table 1). K-means and CAGT (correlation) still recover the count frequency profiles of the two classes with reasonable accuracy at a coverage as low as $f=2$. Note further that probabilistic partitioning is the only method capable of accurately estimating the frequencies of the two classes at a low coverage. This is clearly related to the probabilistic rather than the deterministic assignment of class membership.

3.1.2 Data with flips The next thing we wanted to see was how well the method works when there are flips in the data. We used two classes as before. The simulated data now contain 2000 samples per class, 1000 presented in one orientation and 1000 in the reversed orientation. We compared probabilistic partitioning in shape-based mode to CAGT (correlation) with flipping enabled. Because CAGT (correlation) in default mode returned variable numbers of patterns for $f<5$, we reduced the parameter $k$ to 5...
Table 1. Results for simulated data without shifts or flips

<table>
<thead>
<tr>
<th>Method</th>
<th>( f = 50 )</th>
<th>( f = 10 )</th>
<th>( f = 5 )</th>
<th>( f = 2 )</th>
<th>( f = 1 )</th>
<th>( f = 0.5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-means ( c_1 )</td>
<td>1 (50%)</td>
<td>0.9986 (53.00%)</td>
<td>0.9905 (58.45%)</td>
<td>0.5588 (88.6%)</td>
<td>0.5732 (92.5%)</td>
<td>0.5576 (96.45%)</td>
</tr>
<tr>
<td>K-means ( c_2 )</td>
<td>1 (50%)</td>
<td>0.9999 (47.00%)</td>
<td>0.9993 (41.55%)</td>
<td>0.7443 (11.4%)</td>
<td>0.6459 (7.45%)</td>
<td>0.4590 (3.55%)</td>
</tr>
<tr>
<td>CAGT (Euclidean) ( c_1 )</td>
<td>1 (50%)</td>
<td>1.0000 (49.9%)</td>
<td>0.9990 (50.2%)</td>
<td>0.9742 (59.1%)</td>
<td>0.5370 (92.5%)</td>
<td>0.5802 (96.3%)</td>
</tr>
<tr>
<td>CAGT (Euclidean) ( c_2 )</td>
<td>1 (50%)</td>
<td>1.0000 (50.1%)</td>
<td>0.9998 (49.8%)</td>
<td>0.9950 (40.85%)</td>
<td>0.6459 (7.45%)</td>
<td>0.4965 (3.65%)</td>
</tr>
<tr>
<td>CAGT (correlation) ( c_1 )</td>
<td>1 (50%)</td>
<td>0.9994 (47.9%)</td>
<td>0.9956 (44.53%)</td>
<td>0.9829 (57.06%)</td>
<td>0.5498 (80.6%)</td>
<td>0.5874 (88.3%)</td>
</tr>
<tr>
<td>CAGT (correlation) ( c_2 )</td>
<td>1 (50%)</td>
<td>0.9998 (52.1%)</td>
<td>0.9993 (55.47%)</td>
<td>0.9987 (42.94%)</td>
<td>0.6748 (19.38%)</td>
<td>0.4391 (11.70%)</td>
</tr>
<tr>
<td>Partition ( c_1 )</td>
<td>1 (50%)</td>
<td>1.0000 (49.9%)</td>
<td>1.0000 (50.01%)</td>
<td>0.9998 (50.77%)</td>
<td>0.9985 (51.41%)</td>
<td>0.9862 (51.56%)</td>
</tr>
<tr>
<td>Partition ( c_2 )</td>
<td>1 (50%)</td>
<td>1.0000 (50.03%)</td>
<td>1.0000 (49.99%)</td>
<td>0.9989 (49.23%)</td>
<td>0.9929 (48.59%)</td>
<td>0.9407 (48.44%)</td>
</tr>
</tbody>
</table>

Note: Model accuracy is expressed as Pearson correlation coefficient between original and rediscovered patterns/classes. The percentage of samples attributed to a class is shown in parentheses. The classes \( c_1 \) and \( c_2 \) correspond to the dashed and solid lines in Figure 1, respectively.

Table 2. Classification error (in percentage) between the discovered patterns and their data classes

<table>
<thead>
<tr>
<th>Method</th>
<th>( f = 50 )</th>
<th>( f = 10 )</th>
<th>( f = 5 )</th>
<th>( f = 2 )</th>
<th>( f = 1 )</th>
<th>( f = 0.5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-means</td>
<td>3.00</td>
<td>8.85</td>
<td>40.20</td>
<td>43.85</td>
<td>47.75</td>
<td></td>
</tr>
<tr>
<td>CAGT (Euclidean)</td>
<td>0.30</td>
<td>3.60</td>
<td>32.15</td>
<td>43.85</td>
<td>48.00</td>
<td></td>
</tr>
<tr>
<td>CAGT (correlation)</td>
<td>1.75</td>
<td>5.44</td>
<td>17.41</td>
<td>39.63</td>
<td>43.96</td>
<td></td>
</tr>
<tr>
<td>Partition</td>
<td>0.00</td>
<td>1.05</td>
<td>11.20</td>
<td>23.55</td>
<td>33.95</td>
<td>48.00</td>
</tr>
</tbody>
</table>

Fig. 2. Simulated data with flips. Data (4000 samples) consist of two classes characterized by Gaussian-shaped patterns. Each class is represented by two subsets of 1000 samples, one showing the underlying pattern in native, the other one in reversed (flipped) orientation. Sub-figures \( a1 \), \( b1 \), \( c1 \) and \( d1 \) are for \( f = 50 \) and \( a2 \), \( b2 \), \( c2 \) and \( d2 \) are for \( f = 1 \). \( b1 \) and \( b2 \) are APs of the same data but with all samples presented in their native orientation. It can be seen that the probabilistic partitioning method (shape-based) using flips captures the actual data patterns at a high (\( f = 50 \)) and low (\( f = 1 \)) coverage. The CAGT (correlation) method works well for \( f = 50 \) only. Dashed and solid lines correspond to classes \( c_1 \) and \( c_2 \) in Table 3, respectively.

H3K4me1 and H3K4me3 bin count profiles representing promoter regions to test whether automatic classification methods can correctly identify the two classes of samples and accurately reconstruct the corresponding chromatin signatures (i.e. bin count frequency profiles). As a promoter collection, we used 34,741 annotated TSSs from ENSEMBL. We then extracted H3K4me1 and H3K4me3 tag counts from public ChIP-Seq data for mouse embryonic stem (ES) cells [Creyghton et al., 2010], GEO entries GSM594577 and GSM594581. For each sample, tags for H3K4me1 and H3K4me3 were counted in bins of 50bp over a region of –2500 to +2500 relative the TSS. The two datasets were then combined into one. The advantage of having such a combined dataset (by mixing two real datasets) is that we know the underlying truth, and we can do
the quantitative comparisons similar to what we have done using simulated experiments by trying to separate the two datasets from the combined dataset.

This test dataset potentially poses several new difficulties as compared with the previous synthetic datasets. (i) The two classes are likely to be inhomogeneous themselves because not all promoters are active in ES cells, and this is known to be reflected by the respective histone modification signatures. (ii) The two classes are highly unequal in terms of coverage ($f = 11$ for H3K4me1, $f = 90$ for H3K4me3). This explains why an AP of the mixed dataset looks quasi-identical to an AP for H3K4me3 only (Fig. 3a and b). Because unequal coverage may help to distinguish between the two classes, we tested probabilistic partitioning in both basic- and shape-based mode. (iii) This dataset is much larger than the previously tested synthetic datasets and thus may represent a challenge in terms of CPU requirements. We exploited this fact to carry out a speed comparison of the different programs.

In total, we tested five methods on this dataset, K-means, CAGT (Euclidean), CAGT (correlation), partitioning (basic—non-shape-based) and partitioning (shape-based). The results are shown in Figure 3 and Table 4. Not surprisingly, all methods perform well in reconstructing the H3K4me3 signature around promoters, which dominates the dataset in terms of tag coverage. For the H3K4me1 signature, probabilistic partitioning (shape-based) performs best, followed by CAGT (correlation) and partitioning (basic). A possible explanation for this fact is that coverage is highly inhomogeneous within the H3K4me3 class, causing misclassification of low-coverage H3K4me3 samples as H3K4me1 by the basic but not the shape-based version of probabilistic partitioning. It is noteworthy that CAGT (correlation) outperforms probabilistic partitioning in estimating the relative frequencies of the two classes. This may be due to the fact that CAGT (correlation) was tested on a reduced dataset lacking samples with zeros only.

Regarding speed, we note that probabilistic partitioning (shape-based) is a little slower than CAGT but is still capable of processing the datasets in a few minutes. The speed figures should be interpreted with caution, as they depend on the number of iterations carried out by the probabilistic partitioning algorithm. We further note that K-means is fast but basically incapable of recovering the two histone modification signatures.

### 3.2.2 Application to nucleosome positioning in promoters

In the previous example, we have shown that our method can separate H3K4me1 and H3K4me3 signals that are artificially pooled together. Such a test is useful for method validation but obviously not representative of an interesting biological application. In the following, we apply probabilistic partitioning to a potentially inhomogeneous dataset where the subclasses are not known in advance. Specifically, we analyze the positioning of nucleosomes in human promoters. As anchor points, we use 9714 precisely mapped TSSs from EPDnew version 1 (Drees et al., 2013). Nucleosome mapping data produced by MNase digestion were taken from Schones et al. (2008). Before partitioning, the mapped MNase tags were shifted by 70 bp toward the center of the nucleosome and then counted in bins of 20 bp. Thus, the input data vectors reflect the frequency at which a nucleosome center occurs at a given distance from a TSS. The AP plot for the complete promoter set (Fig. 4) shows a well-positioned +1 nucleosome flanked downstream by a damped oscillatory pattern with the expected period of ∼200 bp. The region immediately upstream of the TSS appears...
Table 4. Model accuracy (represented by Pearson correlation) and classification error between the discovered patterns and their data classes for the various methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>Model accuracy H3K4me1</th>
<th>Model accuracy H3K4me3</th>
<th>Classification error</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-means</td>
<td>0.0244 (83.65%)</td>
<td>0.9980 (16.35%)</td>
<td>33.72</td>
<td>1.16</td>
</tr>
<tr>
<td>CAGT (Euclidean)</td>
<td>0.9270 (69.03%)</td>
<td>0.9987 (30.97%)</td>
<td>23.85</td>
<td>106.31</td>
</tr>
<tr>
<td>CAGT (correlation)</td>
<td>0.9465 (42.86%)</td>
<td>0.9994 (57.14%)</td>
<td>26.98</td>
<td>108.35</td>
</tr>
<tr>
<td>Partition (non-shape-based)</td>
<td>0.8959 (75.76%)</td>
<td>0.9997 (24.24%)</td>
<td>27.26</td>
<td>97.91</td>
</tr>
<tr>
<td>Partition (shape-based)</td>
<td>0.9713 (62.53%)</td>
<td>0.9996 (37.47%)</td>
<td>20.64</td>
<td>149.57</td>
</tr>
</tbody>
</table>

Note: The time (in seconds) taken for each of the methods is also shown. Real data for H3K4me1 and H3K4me3 around TSS regions are mixed (34,741 samples in each dataset with each sample containing 99 bins). The percentage of each class is shown in brackets. H3K4me1 and H3K4me3 stand for the two datasets (Values are rounded to the fourth decimal place for model accuracy and two decimal places for classification error).

to be nucleosome free. No clear oscillatory pattern is seen in the promoter upstream region. The absence of an oscillatory pattern could mean that nucleosomes are randomly positioned or that different promoters have regularly positioned nucleosomes with different phase shifts relative to the TSS. We used shape-based probabilistic partitioning with limited shifting (±1 bins/20 bp) to discriminate between these two alternatives. The results obtained with K = 4 are shown in Figure 4(b–f). With one exception (class 4), the class-specific AP plots show higher nucleosome peaks and stronger oscillatory patterns than the AP plot for the complete set. Therefore, we conclude that the absence of a periodic signal in the upstream region in Figure 4a promoters results from interference of periodic patterns with different phase shifts that almost entirely cancel out each other. We were wondering whether the four promoter classes with distinct nucleosome architectures may differ in terms of regulatory properties.

To this end, we analyzed the distribution of an active and a repressive histone mark (H3K4me3 and H3K27me3) as well as Pol II in the same cell type (Supplementary Fig. S3). We see clear differences. Perhaps most interestingly, classes 2 and 3 show regularly positioned H3K27me3-labeled nucleosomes indicative of a repressed state.

3.2.3 Shape-based peak evaluation with shifting. In this example, we apply probabilistic partitioning to improve a publicly available peak list originating from a ChIP-Seq experiment against a sequence-specific DNA-binding protein. Note that this application is different from the previous ones in that here we are not trying to discover distinct classes. We are merely trying to separate typical examples (belonging to the majority class) from atypical examples, assuming that atypical examples are contaminants. The second goal is to refocus the peak center positions. To reach these objectives, we use shape-based probabilistic partitioning with two classes, one corresponding to the majority class and trained during EM, the other one with a flat count distribution representing background and not modified during EM. As output, we obtain for each peak region in the input list a probability of being a true binding site plus an optimal shifting distance under the true peak model.

To test this approach, we used ChIP-Seq data for CTCF in HUVEC from Broad/ENCODe downloaded from GEO (Barrett et al., 2013). As anchor points, we used the midpoints of the CTCF binding regions given in the peak file included in the GEO sample entry (GSM733716). For each binding region, we counted sequence tags in bins of 10 bp within a 1 kb region around the anchor point. Probabilistic shifting was done by evaluating the ChIP-Seq signal in 31 overlapping windows of 700 bp (70 bins). After partitioning, we split the input peak list into a ‘good’ and a ‘bad’ peak class, applying a threshold probability of 0.5. We also shifted the center positions of the good peaks based on the posterior probability distribution over the 31 shift classes. We then evaluated the peak lists obtained in this way by motif enrichment using the CTCF position weight matrix from the JASPAR database (Portales-Casamar et al., 2010). Figure 5a shows the frequency of CTCF binding motifs around the peak center positions. We note an essentially flat ChIP-Seq signal distribution for the bad peaks and a drastically enhanced Gaussian-like distribution with an increased height and a narrower width for the shifted good peaks. Given the relatively small size of the bad peak set (12,552 of 63,904), the increase in peak height primarily results from shifting and only to a lesser extent from false binding sites elimination.

As a control, we split the same peak list into good and bad examples using the P-values contained in the file downloaded from GEO. (The probability threshold was chosen such as to match the numbers of the subsets obtained with probabilistic
partitioning.) With this filtering criterion, the AP for the bad peak set still shows a low Gaussian-shaped signal distribution, suggesting the retention of a few true binding sites, whereas the good peaks exhibit only a modest increase in signal height (Fig. 5b). The latter was expected because these peaks were not subjected to optimal shifting.

We also evaluated probabilistic peak ranking in terms of reproducibility, using a GEO sample that provides separate peak lists for replicates (see Supplementary Material for details). At an equivalent irreproducible discovery rate of 1%, our method finds slightly fewer peaks than the peak-finder used by the data submitters. However, our peak list was more enriched in CTCF motifs. A possible interpretation of these findings is that our method, which attempts to eliminate peaks of atypical shape, removes artifacts that are reproducibly called by other peak-finders.

Taken together, our results show that probabilistic partitioning is an effective post-processing method for filtering and focusing a publicly available ChIP-Seq peak list obtained with a state-of-the-art peak finder.

3.2.4 Other real examples We wanted to know whether the results obtained with our method would differ from results obtained with another method when applied to the same dataset. Therefore, we tested probabilistic partitioning on an example that was used in (Kundaje et al., 2012) for introduction and illustration of the CAGT algorithm. The details of this analysis are presented in the Supplementary Material. This dataset consists of H3K27ac bin count profiles around CTCF binding sites. Overall, the two methods reveal concordant trends, but the results differ in some details (see Supplementary Material).

4 CONCLUSION

We presented a probabilistic partitioning method to find significant patterns in ChIP-Seq data. The corresponding algorithm runs in $O(n)$ time given a fixed number of classes and EM iterations. It is capable of processing large datasets (tens of thousands of samples) in minutes. The method is conceptually simple yet flexible, and has been implemented in a few lines of R code. The basic partitioning algorithm is readily adjusted to handling flips and shifts following standard principles of EM. With low data coverage, the probabilistic partitioning method gives excellent model accuracy, superior to K-means or CAGT when tested on the same data examples. We have further shown that probabilistic partitioning can serve other purposes than pattern discovery and classification, like partitioning of nucleosome positioning patterns in human promoters, and shape-based evaluation and re-focusing of ChIP-Seq peaks from published peak lists.

Funding: This work was supported by Swiss National Science Foundation (200021_121710/1 to N.U.N., 31003A_125193 to S.K.).

Conflict of Interest: none declared.

REFERENCES


