On Potts Model Clustering, Kernel K-means, and Density Estimation

Alejandro Murua¹, Larissa Stanberry², and Werner Stuetzle²
¹Département de mathématiques et de statistique, Université de Montréal, Canada
²Department of Statistics, University of Washington, Seattle, USA

Key words: Clustering, Potts model, Kernel, K-means, Multiway normalized cut, Density estimation, Monte Carlo, Gene expression.

Abstract

Many clustering methods, such as K-means, kernel K-means, and MNcut clustering, follow the same recipe: (1) choose a measure of similarity between observations; (ii) define a figure of merit assigning a large value to partitions of the data that put similar observations in the same cluster; (iii) optimize this figure of merit over partitions. Potts model clustering, introduced by Blatt, Wiseman, and Domany (1996) represents an interesting variation on this recipe. Blatt et al. define a new figure of merit for partitions that is formally similar to the Hamiltonian of the Potts model for ferromagnetism extensively studied in statistical physics. For each temperature $T$, the Hamiltonian defines a distribution assigning a probability to each possible configuration of the physical system or, in the language of clustering, to each partition. Instead of searching for a single partition optimizing the Hamiltonian, they sample a large number of partitions from this distribution for a range of temperatures. They propose a heuristic for choosing an appropriate temperature and from the sample of partitions associated with this chosen temperature, they then derive what we call a consensus clustering: two observations are put in the same consensus cluster if they belong to the same cluster in the majority of the random partitions. In a sense, the consensus clustering is an “average” of plausible configurations, and we would expect it to be more stable (over different samples) than the configuration optimizing the Hamiltonian.

The goal of this paper is to contribute to the understanding of Potts model clustering and to propose extensions and improvements: (1) We show that the Hamiltonian used in Potts model clustering is closely related to the kernel K-means and MNcut criteria. (2) We propose a modification of the Hamiltonian penalizing unequal cluster sizes and show that it can be interpreted as a weighted version of the kernel K-means criterion. (3) We introduce a new version of the Wolff algorithm to simulate configurations from the distribution defined by the penalized Hamiltonian, leading to penalized Potts model clustering. (4) We note the link between kernel based clustering methods and non-parametric density estimation and exploit it to automatically determine locally adaptive kernel bandwidths. (5) We propose a new simple rule for selecting a good temperature $T$. 
As an illustration we apply Potts model clustering to gene expression data and compare our results to those obtained by model based clustering and a nonparametric dendrogram sharpening method.

1 Introduction

The goal of clustering is to identify distinct groups in a data set and assign a group label to each observation. Clustering is a common problem in emerging fields such as bioinformatics and text mining. In a typical bioinformatics application we may have microarray data measuring the expression levels of thousands of genes for the same organism under different experimental conditions. Genes with similar expression patterns across experiments may have related functions. Clustering of genes can also be a first step towards modeling and understanding gene regulatory networks (Eisen et al. 1998). In text mining, the goal of clustering may be to partition a collection of documents, such as Web pages returned by a search engine, into subsets representing different topics (Tantrum, Murua, and Stuetzle 2002, 2003, 2004).

One of the simplest (and most popular) clustering algorithms is K-means. Let \( x_i \in \mathbb{R}^d, \quad i = 1, \ldots, n \) be our data. Suppose we want to partition the data into \( q \) clusters. Let \( z_{ki} = 1 \) if \( x_i \) belongs to the \( k \)-th cluster, and zero otherwise. K-means finds cluster centers \( \{m_k\}_{k=1}^q \) and cluster memberships \( z_{ki} \) by minimizing

\[
\frac{1}{n} \sum_{k=1}^q \sum_{i=1}^n z_{ki} (x_i - m_k)^t (x_i - m_k).
\]

It is easy to see that this is equivalent to maximizing

\[
\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n <x_i, x_j> \sum_{k=1}^q z_{ki} z_{kj} / n_k,
\]

where \( n_k \) is the number of data points forming the \( k \)-th cluster, \( k = 1, \ldots, q \), and \(<\cdot, \cdot>\) denotes the inner product in \( \mathbb{R}^d \). Define weights \( w(i, j, \{z_{ki}\}) = \sum_{k=1}^q z_{ki} z_{kj} / n_k \). The weight \( w(i, j, \{z_{ki}\}) \) is \( 1/n_k \) if \( x_i \) and \( x_j \) share the same label \( k \), and it is zero otherwise. Using this new notation, we can rewrite the K-means figure of merit as

\[
\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n w(i, j, \{z_{ki}\}) <x_i, x_j>.
\]

As a consequence, one can easily see that (a) K-means penalizes the assignment of the same label to dissimilar data points (\(<x_i, x_j> < 0\)); (b) K-means favors the assignment of the same label to very similar points (large \(<x_i, x_j>\)); and that (c) the effect of the weights \( \{w(i, j, \{z_{ki}\})\} \) is in part to try to assign data points that are not very similar, but still similar (\(<x_i, x_j> > 0\)), to small clusters (small \( n_k \)).

The K-means criterion (1) can be generalized by modifying the weights \( w(i, j, \{z_{ki}\}) \), replacing \(<x_i, x_j>\) with a more general similarity measure \( s(x_i, x_j) \), or both. The criterion (1) then becomes

\[
\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n w(i, j, \{z_{ki}\}) s(x_i, x_j).
\]
We show in Section 2.1 that choosing a similarity measure derived from a Mercer kernel, i.e. 
\[ s(x_i, x_j) = k(x_i, x_j) \] for some square-integrable symmetric positive function \( k : \mathbb{R}^d \to [0, +\infty) \), leads to the kernel K-means criterion (Girolami 2002). An additional modification of the weights results in the Multiway Normalized Cut (MNCut) criterion (see the Appendix).

The figure of merit proposed by Blatt, Wiseman, and Domany (1996a,b, 1997) in their articles introducing what we call Potts model clustering fits into this framework by choosing weights \( \delta_{ij} = \sum z_{ki} z_{kj} \), leading to the criterion

\[
\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} s(x_i, x_j) \delta_{ij}.
\]

The similarity \( s(x_i, x_j) \) between observations \( i \) and \( j \) receives the weight one if they are assigned to the same cluster, and the weight zero otherwise, independent of the cluster sizes. As in K-means, this criterion favors the assignment of the same label to similar points. Clearly, maximizing (3) is equivalent to minimizing

\[
H(\{z_{ki}\}) = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} s(x_i, x_j) - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} s(x_i, x_j) \delta_{ij}
\]

\[
= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (1 - \delta_{ij}) s(x_i, x_j),
\]

since the first term in the equation above does not depend on the label assignments \( \{z_{ki}\} \).

The function \( H(\{z_{ki}\}) \) is just another criterion measuring the quality of clustering, and one could simply find the cluster memberships \( \{z_{ki}\} \) minimizing \( H \). However, Blatt et al. (1996a,b, 1997) pursue a different approach. They point out that \( H(\{z_{ki}\}) \) has a physical interpretation when \( s(\cdot, \cdot) \) is positive and symmetric: it corresponds to the Hamiltonian (Sokal 1996) of a Potts model for describing ferromagnetism phenomena. In this context, the labels are usually referred to as spins. The Potts model is a probabilistic model of the system formed by the particles, i.e. the data points, and their interactions given by the similarity measure. The distribution of the system depends on the temperature \( T \). For each temperature \( T \) there is a probability \( p_T(\{z_{ki}\}) \) associated with each possible configuration of the system’s labels

\[
p_T(\{z_{ki}\}) \propto \exp \left\{ -\frac{1}{T} H(\{z_{ki}\}) \right\} = \exp \left\{ -\frac{1}{2T} \sum_{i=1}^{n} \sum_{j=1}^{n} (1 - \delta_{ij}) s(x_i, x_j) \right\}.
\]

Note that the Potts model gives low probability to configurations with large values of \( H \), i.e. configurations assigning different labels to similar observations.

Blatt et al. (1996a,b, 1997) first simulate a large number \( M \) of configurations \( \{z_{ki}\} \) according to the distribution (6) for a range of temperatures. This can be done efficiently using the Swendsen-Wang Markov Chain Monte Carlo (MCMC) algorithm (Swendsen and Wang 1987, Wang and Swendsen 1990). They propose a heuristic for choosing an appropriate temperature. In a second step they then extract what we call a consensus clustering from the \( M \) configurations associated with the chosen temperature. The consensus clustering assigns two observations to the same cluster if they belong to the same cluster in the majority of the randomly generated configurations. The consensus clusters are the connected components of the graph over the observations with an edge between any pair belonging to the same cluster in the majority of configurations.
In a sense, the consensus clustering is an “average” of plausible configurations, and we would expect it to be more stable (over different samples) than the configuration minimizing $H$. Another great advantage of consensus clustering is that there is no need to specify the number of clusters in the data before starting a search. The number of clusters in a random configuration is itself random and governed by the distribution (6), and forming the consensus does not require any parameters — the clusters and their number are estimated simultaneously. Another advantage is that this approach avoids the combinatorial search for the configuration optimizing $H$. We also noticed in experiments where the true group structure of the data was known that the consensus clustering tends to be closer to the truth than the clustering found by optimizing the figure of merit.

Potts model clustering is also known as the superparamagnetic clustering method. It has been a subject of intensive research since its introduction by Blatt et al in 1996. The physical aspects of the method and its dependence on the definition of the neighbors, type of interactions, number of possible states, and size of the data set has been studied by Wiseman, Blatt and Domany (1998), and by Agrawal and Domany (2003). Ott et al. (2004) introduced a sequential extension to deal with inhomogeneities in shape, density, and size of clusters. Reichardt and Bornholdt (2004) introduced a spin glass Hamiltonian with a global diversity constraint to identify probable community assignments in complex networks. Potts model clustering has been applied to different fields such as computer vision (Domany et al. 1999), gene expression data (Getz et al. 2000, Levine, Domany and Zhang Domany 2003; Einav et al. 2005), high-dimensional chemical data (Ott et al. 2004, 2005) and neuronal spike detection (Quiroga, Nadasdy and Ben-Shaul 2004).

The objective of this paper is to improve and extent Potts model clustering based on statistical methodology and machine learning techniques. More specifically:

1. We show that the Hamiltonian used in Potts model clustering is closely related to the kernel K-means and MNCut criteria. All three criteria are weighted averages of the conditional densities given the cluster labels. The weights differentiate the methods (see Section 2 and the Appendix).

2. We propose a modification of the Hamiltonian penalizing unequal cluster sizes and show that it can be interpreted as a weighted version of the kernel K-means criterion (see Section 3).

3. We introduce a new version of the Wolff algorithm (Wolff 1989) to simulate configurations from the distribution defined by the penalized Hamiltonian, leading to a penalized Potts model clustering (see Section 3.3).

4. We note a link between kernel-based methods and non-parametric density estimation and exploit it to automatically determine kernel bandwidths. While most kernel based clustering methods, including Blatt, Wiseman and Domany’s version of Potts model clustering, use kernels with fixed, pre-determined bandwidth over the entire feature space, our approach produces adaptive bandwidths (Abramson 1982, Silverman 1986) (see Section 4).

5. We propose a simple rule to select a good temperature $T$. Our rule is based on monitoring a series of cluster splitting measures that follow the trajectories over temperature of the cluster sizes. We measure similarity among the clustering partitions generated within and across temperatures by the adjusted rank index (Huber and Arabie, 1985) and its variance. Small variances are indicators of stable partitions and hence possible good partitions. Relevant cluster splitting is also measured through the variation in the upper tail of the distribution of the cluster sizes. The rule proposed by Blatt et al. (1996a,b), namely the variance of the
size of the largest cluster, is a special case of our rule. Our experiments in Section 6 show that our rule performs well.

We apply our proposed Potts model clustering methodology to gene expression data and compare our results to those obtained by model based clustering (Banfield and Raftery 1993, Celeux and Govaert 1995), and the hierarchical clustering with dendrogram sharpening method introduced by McKinney (1995). The former has been shown to perform moderately well for gene expression data (Yeung et al. 2001) when the clustering is done over the genes. However, in many situations the clustering of interest is on the subjects; e.g. being able to differentiate among several subtypes of cancer in order to deliver the optimal treatment. In this case, the data is high-dimensional, with dimensions on the order of $10^4$ genes. Potts model clustering is suitable for this kind of data since the clustering does not depend on the data dimension, but only on the similarities between the data points, and their spatial arrangement. In general, Gaussian model based clustering cannot directly be applied to this type of data, since one would need many more patients than genes in order to estimate the cluster parameters. Throughout our experiments we have observed that Potts model clustering suggested an appropriate number of clusters for the data.

The remainder of the paper is organized as follows. In Section 2 we describe kernel K-means and its connection to Potts model clustering. In Section 3 we study the distribution of labels for different variants of the Potts model and introduce the penalized Wolff algorithm. Section 4 deals with the connection between kernel based methods and kernel density estimation and introduces method for adaptive bandwidth selection. In Section 5 we address the problem of temperature selection for Potts model clustering. In Section 6 we present the results of a simulation performed with the goal of shedding some light into the performance of Potts model clustering and our suggested procedure to select an appropriate temperature. In this section we also apply Potts model clustering to microarray data and illustrate our method for adaptive kernel bandwidth selection. Section 7 contains a discussion and some ideas for future work.

## 2 Connections Between Kernel K-means And Potts Model Clustering

In this section we show that kernel K-means and Potts model clustering are linked to kernel density estimates of the conditional probabilities given the clusters. We also introduce a weighted version of kernel K-means which can be seen as a penalized version of Potts model clustering. As in the Introduction, here we also assume that we know the data consists of $q$ groups.

### 2.1 Kernel K-means

Instead of working directly with the original feature data vectors $x_i$’s, one could work with a suitable transformation of them, say $\Phi : \mathbb{R}^d \rightarrow \mathbb{H}$, where, in general, $\mathbb{H}$ is a higher dimensional (and possible infinite-dimensional) Hilbert space. K-means in this new feature space $\mathbb{H}$ corresponds to finding $z_{ki}$’s and $\mu_k$’s that minimize

$$
\frac{1}{n} \sum_{k=1}^{q} \sum_{i=1}^{n} z_{ki} D(\Phi(x_i), \mu_k) \quad (7)
$$
where \( D(\cdot, \cdot) \) denotes the distance in \( \mathbf{H} \). Obviously, the mean estimates are given by
\[
\hat{\mu}_k = n^{-1}_k \sum_{i=1}^n z_{ki} \Phi(x_i), \quad k = 1, \ldots, q.
\]
Let \( < \cdot, \cdot > \) denote the inner product in \( \mathbf{H} \). Note that
\[
D(\Phi(x_i), \hat{\mu}_k) = <\Phi(x_i), -\hat{\mu}_k> = <\Phi(x_i), \Phi(x_i) > - <\Phi(x_i), \hat{\mu}_k> = - <\hat{\mu}_k, \Phi(x_i)> + <\hat{\mu}_k, \hat{\mu}_k>
\]
Assume that there exist a kernel function in \( \mathbb{R}^d \times \mathbb{R}^d \) for which the inner product in \( \mathbf{H} \) can be expressed as \( <\Phi(x_i), \Phi(x_j)> = k(x_i, x_j) \). In this case K-means does not need to know explicitly the transformation \( \Phi(\cdot) \). It only needs to know the kernel \( k(\cdot, \cdot) \). This is the well-known kernel K-means method (Girolami 2002, Zhang and Rudnicky 2002, Dhillon, Guan and Kulis 2004).

Girolami (2002) shows that the equation (7) can be written as
\[
\frac{1}{n} \sum_{k=1}^q \sum_{i=1}^n z_{ki} k_{ii} - \sum_{k=1}^q \gamma_k R_k,
\]
where \( k_{ij} = k(x_i, x_j) \), \( \gamma_k = n_k/n \) is the proportion of data points falling in cluster \( k \), and \( R_k = n^{-2}_k \sum_{i=1}^n \sum_{j=1}^n z_{ki} z_{kj} k_{ij}, \quad i, j = 1, \ldots, n, \quad k = 1, \ldots, q. \) Since the first term in (8) does not depend on the label assignments (note that \( \sum_{k=1}^q \sum_{i=1}^n z_{ki} k_{ii} = \sum_{i=1}^n (\sum_{k=1}^q z_{ki}) k_{ii} = \sum_{i=1}^q k_{ii} \)), minimizing (7) is equivalent to maximizing
\[
\sum_{k=1}^q \gamma_k R_k = \frac{1}{n} \sum_{i=1}^n \sum_{k=1}^q z_{ki} \frac{1}{n_k} \sum_{j=1}^n z_{kj} k_{ij} = \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^q k_{ij} \sum_{k=1}^q z_{ki} z_{kj} \frac{1}{n_k}
\]
which is exactly the criterion given by (2) with the same weights of the simpler K-means method.
If in addition we assume that \( k(x_i, x_j) = k(x_i - x_j) \) for all \( i, j \) (e.g. Gaussian kernel), then \( \hat{p}(x_i|k) = \frac{1}{n_k} \sum_{j=1}^n z_{kj} k_{ij} \) can be seen as a non-parametric estimate of the conditional density score associated with observing \( x_i \) given cluster \( k \), \( p(x_i|k) \) (Silverman 1986). From now on we will assume that the kernel \( k(\cdot, \cdot) \) is of this form. Therefore (9) can be interpreted as an average of these conditional density scores, and the goal of kernel K-means in this case is to maximize this average.

Girolami (2002) gives a different interpretation to (9). In his view, each \( R_k \) provides a measure of compactness of the corresponding \( k \)-th cluster, \( k = 1, \ldots, q. \) This is derived from the convolution (reproductive-kernel) property of the Gaussian kernel:
\[
\int_{\text{cluster } k} p(x|k)^2 \, dx \approx \int p(x|k)^2 \, dx \approx \int \left( \frac{1}{n_k} \sum_{i=1}^n z_{ki} k(x - x_i) \right)^2 \, dx = \frac{1}{n_k} \sum_{i=1}^n \sum_{j=1}^n z_{ki} z_{kj} k_{ij} = R_k.
\]

### 2.2 Weighted Kernel K-means

On the other hand, \( \gamma_k \hat{p}(x_i|k) \) can be seen as an estimate of the density score associated with observing \( x_i \) in cluster \( k \). Hence
\[
\sum_{k=1}^q \gamma_k^2 R_k = \frac{1}{n} \sum_{i=1}^n \sum_{k=1}^q z_{ki} \gamma_k \frac{1}{n_k} \sum_{j=1}^n z_{kj} k_{ij} = \frac{1}{n} \sum_{i=1}^n \sum_{k=1}^q z_{ki} \gamma_k \hat{p}(x_i|k)
\]
can be interpreted as an average of the density scores associated with observing the data points in the corresponding clusters. This slight modification of the K-means leads to a weighted K-means approach that penalizes the distribution of the cluster sizes. In fact, consider weights given by the $\gamma_k$'s, and the minimization of

$$\frac{1}{n} \sum_{k=1}^{q} \gamma_k \sum_{i=1}^{n} z_{ki} D(\Phi(x_i), \mu_k)$$  \hspace{1cm} (11)

(compare with equation (7)). A straightforward computation, like the one in (Girolami, 2002) leads to the maximization of

$$\sum_{k=1}^{q} \gamma_k^2 R_k - \sum_{k=1}^{q} \gamma_k^2.$$ \hspace{1cm} (12)

The role of the last term above is to penalize non-uniform distribution of the cluster sizes; in other words, its role is to avoid clusters that are too large or too small.

In the next section we show that the criterion given by equation (10) is connected to Potts model clustering. Moreover, we also show that (12) is connected to a modified (penalized) version of Potts model clustering.

### 2.3 Potts Model Clustering

Without loss of generality, assume that the observations are the vertices of a graph. So far we have worked with a complete graph (i.e. all graph nodes are connected). In many practical situations (e.g. images) it may be convenient to work with a reduced graph. For example one can build a K-nearest-neighbor graph such that for each point $x_i$ there is an edge between $x_i$ and its $K$ nearest neighbors. If the K-nearest-neighbor graph contains more than one connected set, then the graph can be augmented by adding edges of the minimum-spanning graph, so that there is a path from any point to any other point in the resulting graph.

Let $\alpha_{ij} = 1$ if $i \neq j$ and points $x_i$ and $x_j$ are neighbors in the graph (i.e. there is an edge connecting these points), and zero, otherwise. A sensible clustering criterion is to penalize different labels between neighboring points. This leads to the minimization of (compare with (5))

$$H(\{z_{ki}\}) = \frac{1}{2} \sum_{i=1}^{n} \sum_{k=1}^{q} \sum_{j=1}^{n} z_{ki}(1 - z_{kj}) k_{ij} \alpha_{ij} = \sum_{(i,j) \text{ neighbors}} k_{ij}(1 - \delta_{ij})$$ \hspace{1cm} (13)

Equation (13) is the Potts model criterion on a graph. As mentioned in the Introduction, it represents the Hamiltonian (Sokal 1996) of the system, which has log-density equal to minus this quantity.

Note that if the graph is a K-nearest-neighbor graph, then the Hamiltonian only involves $O(n)$ terms, whereas for the complete graph in (5) it involves $O(n^2)$ terms. Thus it is computationally attractive and advantageous to work with Potts models on graphs. Although, in general, the graph depends on the inter-point distances themselves, in many interesting situations, such as in images, the graph neighborhood relationship is an intrinsic property of the data. Moreover, as seen in (15) below, working on a graph simply reduces to multiplying the weights $w(i, j, \{z_{ki}\})$ by $\alpha_{ij}$. This holds for every method based on (2) not just for the Potts model clustering.

A trivial calculation yields the following equivalent expression for (13)

$$\sum_{(i,j) \text{ neighbors}} k_{ij}(1 - \delta_{ij}) = \text{constant} - \frac{1}{2} \sum_{j=1}^{n} \sum_{i=1}^{n} \alpha_{ij} k_{ij} \delta_{ij}.$$ \hspace{1cm} (14)
The constant in the right-hand-side of (14) is independent of the labels. Therefore, maximizing the likelihood of the Potts model (i.e. minimizing (13)), excluding the trivial all-in-one cluster solution, is equivalent to maximizing

\[
\frac{1}{2} \sum_{j=1}^{n} \sum_{i=1}^{n} \alpha_{ij} k_{ij} \delta_{ij} = \frac{1}{2} \sum_{j=1}^{n} \sum_{k=1}^{q} \sum_{i=1}^{n} z_{ki} z_{kj} k_{ij} \alpha_{ij}
\]  

(15)

Now, note that \( \hat{p}_{potts}(x_i|k) = n_k^{-1} \sum_{j=1}^{n} z_{kj} k_{ij} \alpha_{ij} \) is an estimate of \( p(x_i|k) \). Thus, we can re-write (15) as

\[
\frac{n}{2} \sum_{i=1}^{n} \sum_{k=1}^{q} z_{ki} \gamma_k \hat{p}_{potts}(x_i|k)
\]  

(16)

Therefore, the equation (15) is equivalent to the equation (10) and it can be interpreted in the same manner.

2.4 The Connection With Weighted K-means

Adding the term \( (n^2/2) \sum_{k=1}^{q} \gamma_k^2 \) to the expression in (15) leads to an expression similar to (12), derived from the weighted K-means criterion. We refer to this latter model as the penalized Potts model. As in weighted K-means, the distribution of the cluster sizes are shrunk towards the uniform distribution (equal size clusters). It is easily shown (see Section 3.3 below) that both criteria are exactly the same for the complete graph (i.e., \( \alpha_{ij} = 1 \) for all \( i, j = 1, \ldots, n \)). When the graph is a reduced graph (e.g. K-nearest-neighbor graph) the criteria differ. Of course, from a computational point of view it is advantageous to use small neighborhoods with penalized Potts model clustering. Below in Sections 3.3 and 3.4, we develop an extended Potts model and a “penalized” Wolff algorithm with the aim of optimizing this criterion.

3 Simulating The Labeling Distribution

A great advantage of the Potts model clustering method over other kernel-based clustering methods is that it can estimate the clusters and their number simultaneously. Cluster membership is based on the proportion of times that any two observations are assigned to the same cluster. These proportions are estimated using MCMC techniques such as the Swendsen-Wang algorithm (Swendsen and Wang 1987), or the Wolff algorithm (Wolff 1989). For completeness, we briefly outline the procedure here.

3.1 The Swendsen-Wang And Wolff Algorithms

The clusters are found through a MCMC simulation. We explain here the Swendsen-Wang and Wolff algorithms. This exposition is needed in order to introduce the penalized Wolff algorithm in Section 3.3.

Perhaps the simplest way to generate samples from the Potts model is through a Gibbs sampler (Grenander 1983, Geman and Geman 1984), also known in physics as the heat bath algorithm (Creutz 1979), on the labels \( \{z_{ki}\} \). This reduces to finding the full conditionals of each vector \( (z_{1i}, z_{2i}, \ldots, z_{qi}) \) given the current value of the remaining labels for \( j \neq i \), for \( i = 1, \ldots, n \). Although the conditionals are easy to obtain and work with, the sampling is rather inefficient. For example, to assign two points, say \( x_i \) and \( x_j \), to the same label may take a full sweep of the data, let alone assigning several points to the same updated label. Fortunately, there exists a very efficient
way to generate samples from the Potts model by model augmentation. Let \( p_{ij} = 1 - \exp\{-k_{ij}\} \).

The Potts model density is given by

\[
p(\{z_{ik}\}) = Z^{-1} \exp\left(- \sum_{(i,j) \text{ neighbors}} k_{ij}(1 - \delta_{ij})\right)
\]

\[
= Z^{-1} \prod_{(i,j) \text{ neighbors}} \left(1 - p_{ij} + p_{ij}\delta_{ij}\right)
\]

where \( Z = \sum_{\{z_{ki}\}} \exp\{-H(\{z_{ki}\})\} \) is the corresponding normalizing constant. Following Sokal’s derivation (Sokal 1996), since the sum of any two real numbers \( x, y \), can be written as \( x + y = \sum_{b=0}^{1} x(1-b) + yb \), it follows that

\[
Z = \sum_{\{z_{ki}\}} \sum_{\{b_{ij}\}} \prod_{(i,j)\text{neighbors}} \{(1 - p_{ij})(1 - b_{ij}) + p_{ij}b_{ij}\delta_{ij}\},
\]

where the \( \{b_{ij}\} \) are binary \( 0 - 1 \) variables. They are said to be the bonds between the vertices of the graph generated by the data. This formula leads to a joint density of labels and bonds

\[
p(\{z_{ki}\}, \{b_{ij}\}) = Z^{-1} \prod_{(i,j)\text{neighbors}} \{(1 - p_{ij})(1 - b_{ij}) + p_{ij}b_{ij}\delta_{ij}\},
\] (17)

which is known as the Fortuin-Kasteleyn-Swendsen-Wang model (Sokal 1996). The marginal density over the labels is exactly the Potts model. The marginal over the bonds is known as the random-cluster model. The interpretation of the bond variables in model (17) is the following. The bond \( b_{ij} \) is said to be frozen if \( b_{ij} = 1 \), and the points \( x_i \) and \( x_j \) are neighbors \( (\alpha_{ij} = 1) \) and have the same label \( (\delta_{ij} = 1) \). Otherwise, the bond is not frozen: \( b_{ij} = 0 \). Again according to (17), the bond \( b_{ij} \) becomes frozen with probability \( p_{ij} = 1 - \exp\{-k_{ij}\} \). A set for which any two points can be connected by a path of frozen bonds is said to be a connected set. Note that only subsets containing points with the same label can form a connected set.

The Swendsen-Wang algorithm uses (17) to generate samples from the Potts model via MCMC simulation. This is a Gibbs sampler with two steps:

**Step 1.** Given the labels \( \{z_{ki}\} \), each bond becomes frozen independently of the others with probability \( p_{ij} \) if \( \alpha_{ij} = 1 \) and \( \delta_{ij} = 1 \); otherwise the bond is set to 0.

**Step 2.** Given the bonds \( \{b_{ij}\} \), each connected subset is assigned the same label. The assignment is done independently and chosen uniformly at random from the set of labels \( \{1, 2, \ldots, q\} \).

Note that the connected sets formed by frozen bonds allow for cluster splitting (Step 1). Merging is produced by the label assignment (Step 2). In the Swendsen-Wang algorithm both merging and splitting are done in parallel, since a multitude of sites in the graph are updated simultaneously in each iteration of the algorithm.

The Wolff algorithm (Wolff 1989) is a variant of the second step above. Instead of updating all connected sets, a point in the graph is chosen uniformly at random; the associated connected set is then updated as in the Swendsen-Wang algorithm. The advantage of this variant is that larger clusters are updated more often.

9
3.2 The Clusters

Several (simulated) samples drawn from the Potts model are collected. The cluster structure is then estimated from these samples by counting how many times any two given points are given the same label. Hence the label assignments are based on MCMC estimates $\hat{Q}_{ij}$ of the probabilities (under the Potts model)

$$Q_{ij} = p(z_{ki} = z_{kj} \text{ for some } k \in \{1, \ldots, q\}) = p(\delta_{ij} = 1).$$

If $\hat{Q}_{ij}$ is larger than a certain threshold (usually 0.5), then points $x_i, x_j$ are assigned to the same cluster.

3.3 Penalized Potts Model Clustering

Penalized Potts model clustering aims at maximizing (see the right-hand-side of (15))

$$\frac{1}{2} \sum_{j=1}^{n} \sum_{i=1}^{n} \alpha_{ij} k_{ij} \delta_{ij} - \frac{n^2}{2} \sum_{k=1}^{q} \gamma_k^2. \quad (18)$$

It is useful to note that maximizing this criterion is the same as maximizing

$$\frac{1}{2} \sum_{j=1}^{n} \sum_{i=1}^{n} \alpha_{ij} k_{ij} \delta_{ij} - \frac{n^2}{2} \sum_{k=1}^{q} \left(\gamma_k - \frac{1}{q}\right)^2;$$

hence the penalty term tends to balance the sizes of the clusters, as mentioned before. Next, note that $\sum_{i=1}^{n} z_{ki} = n_k$. One can rewrite the last term as

$$n^2 \sum_{k=1}^{q} \gamma_k^2 = \sum_{k=1}^{q} n_k^2 = \sum_{k=1}^{q} \sum_{j=1}^{n} \sum_{i=1}^{n} z_{ki} z_{kj} = \sum_{j=1}^{n} \sum_{i=1}^{n} \alpha_{ij} \delta_{ij} + \sum_{j=1}^{n} \sum_{i=1}^{n} (1 - \alpha_{ij}) \delta_{ij}. \quad (19)$$

With the penalty criterion, it is convenient to rewrite $k_{ij}$ as $k_{ij} + 1$. Then, using the identity in (19), the penalized criterion (18) can be written as

$$\frac{1}{2} \sum_{j=1}^{n} \sum_{i=1}^{n} \alpha_{ij} k_{ij} \delta_{ij} - \frac{1}{2} \sum_{j=1}^{n} \sum_{i=1}^{n} (1 - \alpha_{ij}) \delta_{ij}. \quad (20)$$

Therefore, in addition to penalizing the assignment of different labels to neighboring points, penalized Potts model clustering also imposes a penalty whenever non-neighboring points are assigned the same label.

3.4 Simulating The Labeling Distribution: The Penalized Wolff Algorithm

We have developed a variant of the Wolff algorithm to estimate the cluster structure under the criterion (18). The key idea is to note that the last term in (20) can be seen as a requirement (penalty) on the graph formed by connecting all non-neighboring points. Hence, one can augment the model by introducing bonds between non-neighboring points in the same manner as bonds were introduced in the original graph in the Swendsen-Wang algorithm.

Let $\{d_{ij} \in \{0, 1\}\}$ be the set of non-neighbor bonds. A non-neighbor bond $d_{ij}$ can become frozen only if $x_i$ and $x_j$ are not neighbors, and $\delta_{ij} = 0$. In this latter case $d_{ij}$ becomes frozen with probability $q_{ij} = 1 - e^{-1}$. 

10
We say that there is a connected path between points \( x_i \) and \( x_j \) if there exist a path of consecutive frozen-bond edges that starts at \( x_i \) and finishes at \( x_j \). The connected path of a point \( x_i \) is the set of points connected to \( x_i \) by a connected path.

The penalized Wolff’s algorithm works as follows

1. Given the labels \( \{ z_{ki} \} \) and the non-neighbor bonds \( \{ d_{ij} \} \), set each bond \( b_{ij} \) to 1 independently of other bonds with probability \( p_{ij} = 1 - e^{-k_{ij}} \) if the following four conditions hold: (i) \( \alpha_{ij} = 1 \), (ii) \( \delta_{ij} = 1 \), (iii) there is no non-neighbor frozen-bond between the point \( x_j \) and another point in the connected path of \( x_i \), and (iv) there is no non-neighbor frozen-bond between the point \( x_i \) and another point in the connected path of \( x_j \). Otherwise set the bond to 0.

2. Given the labels \( \{ z_{ki} \} \) and the bonds \( \{ b_{ij} \} \), set each non-neighbor bond \( d_{ij} \) to 1 independently of other non-neighbor bonds with probability \( q_{ij} = 1 - e^{-1} \) if the following three conditions hold: (i) \( \alpha_{ij} = 0 \), (ii) \( \delta_{ij} = 0 \), and (iii) there is no connected path between points \( x_i \) and \( x_j \). Otherwise set the non-neighbor bond to 0.

3. Given the bonds \( \{ b_{ij} \}, \{ d_{ij} \} \), choose a point \( x \) uniformly at random. Find the associated connected subset \( A = A(x) \), and the associated set \( B(A) \) of non-neighbor points that have a non-neighbor frozen bond with at least one of the points in the connected subset \( A \). From the set \( C(B) \) of all labels associated with points in \( B(A) \), and its complement \( \overline{C(B)} \). This latter set is the set of admissible labels. Choose a label uniformly at random from the set of admissible labels. Assign this label to all points in \( A \).

The final cluster structure is estimated as explained in Section 3.2.

4 The Connection With Density Estimation

Equation (16) connects Potts model clustering with density estimation. The interaction terms \( k_{ij} \) can be thought of as the contribution of the point \( x_j \) when evaluating the kernel density at the point \( x_i \). This interpretation of Potts model clustering leads to some improvements of the model as shown below. By analogy with kernel density estimation, one could use an adaptive bandwidth in the interaction terms. Using the quick estimate of the density at \( x_i \), \( \hat{p}_{knn}(x_i) \), obtained at the time the K-nearest-neighbor graph for the data is constructed (see the beginning of Section 2.3), we derive a localized bandwidth (Breiman, Meisel and Purcell 1977, Abramson 1982, Silverman 1986 Section 5.3)

\[
\lambda_{knn}(x_i) \propto \exp \left\{ -\frac{1}{2} \left( \log[\hat{p}_{knn}(x_i)] - \frac{1}{n} \sum_{j=1}^{n} \log[\hat{p}_{knn}(x_j)] \right) \right\} \tag{21}
\]

Since the Potts model uses a symmetric kernel, we symmetrized the adaptive bandwidth kernel by replacing \( k_{ij} \) with

\[
k_{ij}^{(s)} = \frac{1}{2} \left( k(\lambda_{knn}^{-1}(x_i)(x_i - x_j)) + k(\lambda_{knn}^{-1}(x_j)(x_j - x_i)) \right)
\]

In our experiments, this choice of bandwidth often improved the estimation of the clustering structure in the data. We refer to the algorithm run with these bandwidths as the adaptive Potts model clustering algorithm.
4.1 Bandwidth Estimation

The adaptive bandwidth given by the equation (21) can be used as a starting value for the procedure that estimates the local bandwidths and the clustering structure simultaneously. The outcome of such a procedure would be the clustering structure and a kernel density estimate of the density associated with the data.

The MCMC machinery already in place to estimate the clusters can be extended to estimate the bandwidths. For example, a simple prior for the bandwidth at point \( x_i, \lambda_i \), is an inverse Chi-squared distribution with scale given by the adaptive bandwidth in (21), \( i = 1, \ldots, n \), and a few degrees of freedom. The modified algorithm has two steps

1. For given bandwidths \( \{\lambda_i\} \), update the labels as in the Swendsen-Wang, Wolff, or penalized-Wolff algorithm.

2. For given labels, update the bandwidths independently of each other through a Metropolis-Hastings procedure (Metropolis 1953, Hastings 1970).

In order to account for the effect of the bandwidths, one could update the labels several times before attempting to update the bandwidths. Obviously, one also could update the bandwidths several times before updating the labels.

In what follows we describe the Metropolis-Hastings sampler for the bandwidth with an inverse Chi-squared prior. Recall that for a given target density \( \pi(\lambda) \) and proposal density \( q(\lambda^*|\lambda) \), the Metropolis-Hastings algorithm proceed as follows: given the current state \( \lambda \), an update to state \( \lambda^* \) is proposed with density \( q(\lambda^*|\lambda) \); the update is accepted with probability

\[
A(\lambda^*, \lambda) = \min \left\{ 1, \frac{q(\lambda|\lambda^*)\pi(\lambda^*)}{q(\lambda^*|\lambda)\pi(\lambda)} \right\}.
\]

In our particular case the joint density of labels and bandwidths is proportional to

\[
\pi(\lambda) = \prod_i \lambda_i^{(\nu+1)/2} \exp\left\{-\frac{1}{2} \frac{\nu s^2_i}{\lambda_i} \right\} \times \exp\left\{-\frac{1}{2} \sum_{(i,j) \text{ neighbors}} k^{(s)}_{ij}(\lambda_i, \lambda_j)(1 - \delta_{ij}) \right\},
\]

where \( s^2_i \) are the prior scales, and \( \nu \) is the prior degrees of freedom. We have used the notation \( k^{(s)}_{ij}(\lambda_i, \lambda_j) \) to make explicit the dependency of the symmetrized kernel on both bandwidths \( \lambda_i \) and \( \lambda_j \). At each location \( x_i \), consider an inverse Chi-squared proposal density \( q(\lambda^*_i|\lambda_i) \) with scale

\[
s^2_i + \frac{1}{\nu} \sum_{(i,j) \text{ neighbors}} \lambda_i k^{(s)}_{ij}(\lambda_i, \lambda_j)(1 - \delta_{ij}),
\]

and \( \nu \) degrees of freedom. Then the acceptance ratio for the proposal is

\[
R(\lambda^*_i, \lambda_i) = \exp\left\{-\frac{1}{2} \sum_{(i,j) \text{ neighbors}} (1 - \delta_{ij}) [k^{(s)}_{ij}(\lambda^*_i, \lambda_j) - k^{(s)}_{ij}(\lambda_i, \lambda_j)] \right\}
\]

and we accept the update \( \lambda^*_i \) with probability \( A(\lambda^*_i, \lambda_i) = \min\{1, R(\lambda^*_i, \lambda_i)\} \).

If bandwidths associated with nearby points are expected to be similar, then the above procedure might not be adequate. Instead some penalty prior on the smoothness of the bandwidths can be used. We experimented with two priors on the log-bandwidths, a Gaussian prior, and a Laplace-type prior. As expected, the Gaussian prior yields smoother bandwidths, whereas the Laplace prior yields piecewise-constant looking bandwidths (see Section 6.3). Despite the difference in bandwidth estimates, both the Gaussian and Laplace priors give similar smooth estimates of the underlying kernel density, at least in the one and two-dimensional examples described in Section 6.3. Next, we described the implementation of these two smoothing priors.
4.2 Smoothing Priors

Let $\tau_i = \log \lambda_i$, $i = 1, \ldots, n$. The Gaussian smoothing prior for the bandwidths has the form

$$p_n(\{\tau_i\} | \{z_{ki}\}) \propto \exp\left\{ \frac{-1}{2\sigma^2} \sum_{i=1}^{n} \sum_{x_j \text{ neighbor of } x_i} (\tau_i - \tau_j)^2 \delta_{ij} \right\} \times \exp\left\{ \frac{-1}{2\sigma_0^2} \sum_{i=1}^{n} (\tau_i - \log \lambda_{knn}(x_i))^2 \right\}. $$

Hence only bandwidths of neighboring points with the same label are expected to be similar. The variance $\sigma^2$ acts as a penalty cost. As before, bandwidth updates are generated using a Metropolis-Hastings sampler. Our proposal $\tau^*_i$ is generated from the Gaussian density with mean $\mu_i$ and variance $\sigma_i^2$ given by

$$\mu_i = \sigma_i^2 \left( \frac{2}{\sigma^2} \sum_{x_j \text{ neighbor of } x_i} \tau_j \delta_{ij} + \frac{1}{\sigma_0^2} \log \lambda_{knn}(x_i) \right)$$

$$\sigma_i^2 = \left( \frac{2 m_i}{\sigma^2} + \frac{1}{\sigma_0^2} \right)^{-1}$$

where $m_i$ is the number of neighbors of $x_i$ with the same label as $x_i$. With this proposal, the acceptance ratio $R_2(\tau^*_i, \tau_i)$ is given by

$$R_2(\tau^*_i, \tau_i) = \exp\left\{ -\frac{1}{2} \sum_{x_j \text{ neighbor of } x_i} [k_{ij}^{(s)}(\lambda^*_i, \lambda_j) - k_{ij}^{(s)}(\lambda_i, \lambda_j)](1 - \delta_{ij}) \right\}.$$ 

Similarly, our Laplace-type prior has the form

$$p_l(\{\tau_i\} | \{z_{ki}\}) \propto \exp\left\{ \frac{-\Delta}{2} \sum_{i=1}^{n} \sum_{x_j \text{ neighbor of } x_i} |\tau_i - \tau_j| \delta_{ij} \right\},$$

where $\Delta$ is the penalty cost parameter. In this case, our proposal $\tau^*_i$ is generated from the Laplace density with location equal to the median of the bandwidths $\tau_i$ and $\tau_j$’s associated with neighboring points with the same label as $x_i$. Let $\tilde{\mu}_i$ denote this median, and $\tilde{\mu}^*_i$ denote the median of the bandwidths $\tau^*_i$ and $\tau_j$’s associated with neighboring points with the same label as $x_i$. The acceptance ratio $R_1(\tau^*_i, \tau_i)$ is given by

$$R_1(\tau^*_i, \tau_i) = \exp\left\{ -\Delta ||\tau_i - \tilde{\mu}_i^*| - ||\tau^*_i - \tilde{\mu}_i||\right\} \times \exp\left\{ -\Delta \left[ \sum_{x_j \text{ neighbor of } x_i} (|\tau^*_i - \tau_j| - |\tau_i - \tau_j|) \delta_{ij} \right] \right\} \times \exp\left\{ -\frac{1}{2} \sum_{x_j \text{ neighbor of } x_i} [k_{ij}^{(s)}(\lambda^*_i, \lambda_j) - k_{ij}^{(s)}(\lambda_i, \lambda_j)](1 - \delta_{ij}) \right\}.$$ 

4.3 Density Estimation

Recall, that through MCMC one gets an estimate of the posterior distribution given the data for each of the bandwidths. Let $\lambda_i$ be the median bandwidth at $x_i$, estimated from one of the procedures outlined in the previous subsections. Let $p(x | k) = n_k^{-1} \sum_{i=1}^{n} z_{ki} k([x - x_i] / \lambda_i)$. The density estimator at $x$ is $\hat{f}(x) = \sum_{k=1}^{q} \gamma_k p(x | k) = \frac{1}{n} \sum_{i=1}^{n} k([x - x_i] / \lambda_i)$. Section 6.3 gives an idea of how this estimate works.
5 Temperature Selection

An important parameter in the Potts model is the temperature, \( T \). Recall from the introduction that this enters as a denominator in the weights \( w(i, j, \{ z_{ki} \}) = 1/T \) (see (2)), and hence as a denominator of the Hamiltonian \( H(\{ z_{ki} \})/T \). Although at first sight the temperature does not seem to influence the criterion given by (2), it does have an important effect on the label assignments drawn (sampled) from the Potts model. As before, at any given temperature, the cluster structure is estimated from the simulated label samples by counting how many times any two given points are assigned the same label. As seen in Section 3.2, the label assignments are based on the probabilities \( \{ Q_{ij} = Q_{ij}(T) \} \). It turns out that these probabilities are directly related to the number of times any two given points occur in the same connected subset, and hence to probabilities under the random clusters model resulting from integrating out the labels in the joint density of labels and bonds (17) (Edwards and Sokal 1988). The log-density of the random cluster model for a given temperature \( T \) is proportional to

\[
\sum_{(i,j) \text{ neighbors}: b_{ij}=1} \log(1 - e^{-k_{ij}/T}) - \frac{1}{T} \sum_{(i,j) \text{ neighbors}: b_{ij}=0} k_{ij} + C(\{ b_{ij} \}) \times \log(q),
\]

where \( C(\{ b_{ij} \}) \) denotes the number of connected components given the current values of the bonds. Note that this function favors more clusters when \( T \) is large, and fewer clusters when \( T \) is small. Since the Potts model cluster structure is given by the connected subsets favored by the random clusters model (25), \( T \) controls the number of connected subsets and their structure. That is, \( T \) acts as a clustering smoothing parameter. Note that the value of \( q \), i.e. the original number of labels, has little relevance unless it is very large. In this latter case the model would favor a large number of connected subsets. By varying the temperature from low to high values, Potts model clustering can be seen as a hierarchical splitting procedure. Thus the key to Potts model clustering is to find the “right” temperature associated with the “true” group structure in the data. This is a hard problem and more research is needed to solve it. The current strategy is to try several values of the temperature parameter and monitor some temperature-dependent statistics of the “goodness-of-clustering”. Based on the physical model underlying the Potts model density, Blatt et al. (1996a,b) suggested monitoring the magnetization of the system to obtain clues on which temperatures produce cluster splitting. In statistical terms, this measure is the variance of the size of the largest cluster. In our experiments we noted that this is not all the time a good measure of splitting, since smaller clusters might split before the larger ones. Hence an extension of the magnetization is to monitor the variance of the size of (possible many of ) the largest clusters. The number of clusters to monitor depends on the number of clusters one expect to observe. Peaks on these variances indicate important splits in the current cluster structure. Since it is assumed that the true but unknown cluster structure in the data corresponds to a temperature nearby one of these peaks, one could travel over the temperature range rather quickly in order to get some rough estimates of the temperatures where peaks occur. Once peaks are located, longer simulations could be run at these peak temperatures in order to study in detail the cluster structures that they yield.

5.1 Choosing A Good Temperature

In our experiments we used a simple rule to select the temperature which appears to uncover the best clustering structure. We note that monitoring cluster splitting is a way to measure variation in clustering structures or partitions of the data. Hence our rule is based on two approaches
to measure the variation in clustering structures: (a) the distribution of the size of the largest clusters (an extension of the magnetization measure); and (b) the adjusted Rand index (Rand 1975, Milligan and Cooper 1986, Huber and Arabie, 1995). These are explained next.

(a) The distribution of the size of the largest clusters. For any given \( T \), let \( C_\ell(T) \) be the \( \ell \)-th largest cluster found in a partition (sample) drawn from the Potts model, and \( S_\ell(T) \) be its corresponding cluster size, \( \ell = 1, \ldots, G \), \( S_1(T) \geq S_2(T) \geq \cdots \geq S_G(T) \). \( G \) is a parameter of the procedure that depends on our prior belief on the true number of clusters in the data. In our experiments we set \( G = 6 \). We monitor the curves (on \( T \))

\[
\text{MVar}(L, G, T) = \frac{1}{G - L + 1} \sum_{\ell=L}^{G} \text{Var}(S_\ell(T)) \quad L = 1, \ldots, G - 1,
\]

and choose a clustering associated with a temperature immediately following one of the local maxima of the curves \( \text{MVar}(L, G, T) \), \( L = 1, \ldots, G - 1 \). Usually all these curves peak around the same temperatures, so the choice of \( L \) is not very relevant. However, we do recommend using \( L > 1 \), since the largest cluster is usually very large in comparison with the other clusters, and hence its variance masks the variance of the size of the other clusters.

The adjusted Rand index. There are many known measures for comparison of partitions such as the Folkes-Wallace index and the adjusted Rand index (Milligan and Cooper 1986, Huber and Arabie, 1995) that are popular in the clustering literature (Yeung et al. 2001, Tantrum et al. 2004). The Rand index (Rand 1971) is the fraction of pairs of points that are either in the same clusters in both partitions or in different clusters in both partitions. The adjusted Rand index adjusts the Rand index so that its expected value is zero when the partitions are random. The larger the adjusted Rand index, the more similar the two partitions are. We monitor the similarity among the partitions sampled by the Potts model at each temperature with the adjusted Rand index as follows. Suppose that at each temperature \( T \), \( M \) partitions \( \{P_m\}_{m=1}^M \) are generated from the Potts model density. A representative partition is given by the consensus clustering among the \( M \) partitions \( P(T) \). Let \( r_m \) be the adjusted Rand index between \( P_m \) and \( P(T) \). The average Rand index \( \bar{r}(T) \) is a measure of similarity among the \( M \) partitions, whereas \( \text{Var}(\bar{r}(T)) = \frac{\sum_{m=1}^M (r_m - \bar{r}(T))^2}{(M - 1)} \) is a measure of instability. Hence in principle a good temperature \( T_0 \) is a temperature for which \( \bar{r}(T_0) \) is high and \( \text{Var}(\bar{r}(T_0)) \) is low. We also expect that the consensus partitions associated with temperatures in a neighborhood of \( T_0 \) be similar to the consensus partition found at \( T_0 \). In other words, the system induced by the Potts model should be more or less stable at temperatures close to \( T_0 \). An important implication of this latter requirement is that the choice of \( T_0 \) should not be too critical as long as it is chosen in the neighborhood of the (unknown) optimal temperature. The similarity between two consensus partitions \( P(T - \Delta T) \) and \( P(T) \) generated by consecutive temperatures \( T - \Delta T \), and \( T \) is measured by their adjusted Rand index \( R(T) \).

The rule to select \( T_0 \). We select the first candidate temperature \( T_0^\ast \) located at the end of the last significant maximum of \( \text{MVar}(L, G, T) \) that precedes a sharp drop in the cluster-size variation. Then we check for temperatures near \( T_0^\ast \) that have high \( \bar{r}(T) \), low \( \text{Var}(\bar{r}(T)) \), and are found in a more or less stable region of \( R(T) \) (i.e. a plateau of \( R(T) \)). The final choice \( T_0 \) is a compromise between all these conditions. Figure 1 illustrates the procedure. Our experiments in the next section show that this procedure performs well.
Figure 1: Plots of the adjusted Rand index across temperatures, $R(T)$ (upper left corner), mean adjusted Rand index within temperatures, $\bar{r}(T)$ (upper right corner), standard deviation of the adjusted Rand index within temperatures, $\sqrt{\text{Var}(r(T))}$ (lower right corner), and mean standard deviations, $\sqrt{\text{MV}_{\text{ar}}(L, G = 6, T)}$, $L = 2, \ldots, 6$ (lower left corner). The vertical line corresponds to the chosen temperature $T_0 = 0.10$. The statistics were computed for the artificially generated data set $\mathcal{D}_2$ based on 3000 partitions kept after a burn-in period of 300.

6 Experimental Results

6.1 Performance On Simulated Data

In this section we report the results of a simulation carried out to study the performance of Potts model clustering on different data sets and for different values of the Swendsen-Wang simulation parameters. To assess the performance, the data used in this section were artificially generated. These are three two-dimensional data sets: $\mathcal{D}_1$ consisting of 200 points in two clusters, $\mathcal{D}_2$ consisting of 200 points in four clusters, and $\mathcal{D}_3$ consisting of 340 points in eight clusters. The data are plotted in Figure 2. The clusters are either Gaussian clumps or uniformly scattered around arcs.

The Adjusted Rand Index

The goal of this study is to find the right combination of the burn-in and partitions kept (after the burn-in period) parameters of the Swendsen-Wang algorithm (SWA) within the Potts model clustering procedure. The burn-in parameter corresponds to the number of initial partitions generated by the SWA that are discarded from further analysis. The partitions generated by the SWA after the burn-in are the partitions kept to compute the consensus clustering and all relevant statistics. In our simulation, the burn-in and partitions-kept parameters were set to values in \{100, 300, 600, 1000, 3000\}. We ran Potts model clustering on each data set $\mathcal{D}_i$ ($i = 1, 2, 3$) with all 25 combinations of the burn-in and partitions-kept parameters. For each combination we measured the number of clusters associated with the partition chosen according to the procedure...
explained in Section 5.1, and the adjusted Rand index between the true partition and the chosen partition. The results are summarized in Figure 3. It appears that the performance of Potts model clustering is not sensitive to the burn-in parameter. It is also clear that the performance with only 100 partitions-kept is very different and much poorer than the performance with at least 300 partitions-kept. The analysis of variance on the resulting adjusted Rand indexes confirmed these observations. An HSD (honestly significant difference) Tukey method for all pairwise comparisons (Tukey 1949) at a significance $\alpha = 0.01$ revealed no significant difference between the performance of Potts model clustering with 300, 600, 1000 or 3000 partitions-kept. However the analysis of variance on the resulting difference between the square-roots of the estimated number of clusters and the true number of clusters showed a significant interaction between the data sets and the number of partitions kept. As observed from Figure 3, the performance with at most 300 partitions-kept is worse than the performance with at least 600 partitions-kept for the data set $D_3$, but not for the other data sets. This indicates that the appropriate number of partitions-kept is dependent on and increasing with the complexity of the data.

The Number Of Clusters

Potts model clustering with the temperature choice described in Section 5.1 yielded the correct number of clusters 55% of the time. 88% of the time the estimated number of clusters was off by at most by one cluster; 92% of the time it was within three clusters. As observed in the previous section, increasing the number of partitions kept increased the proportion of correct estimates of the number of clusters. In fact, the proportion of times the algorithm estimated the correct number of clusters when 300 partitions were kept after burn-in was 0.33; the proportion increased to 0.53, 0.60, and 0.73 when the number of partitions kept after burn-in was increased to 600, 1000 and 3000, respectively. We note that the results of this section were based on the simulations described in the previous section with the exception of the simulations performed with only 100 partitions kept, since we have concluded from our analysis above that these were significantly different from the remaining simulations.
The conclusion of this simulation study is that a small burn-in and a moderate data-dependent number of partitions kept (e.g., about 1000) were enough to obtain a good performance. Although there was no evidence that keeping several thousands partitions would improve the adjusted Rand index in a significant way, keeping a large number of partitions would probably improve the estimation of the number of clusters. We note that our procedure always starts at a cool temperature, i.e., a temperature for which no significant splitting is observed, and proceeds with a hotter temperature with the initial partition equal to the last partition drawn from the previous (colder) temperature. Hence, even if we set a small burn-in, the partitions drawn at hot temperatures have been drawn after a large “burn-in period” consisting of all previous draws at colder temperatures. In other words, the initial partition fed to Potts model clustering at any given temperature (other than the coldest ones) is likely to be a very good starting point for the Swendsen-Wang or Wolff algorithms. Hence, the results of the simulation indicating that there is no significant difference in adjusted Rand index if the SWA is run with 300 or 3000 partitions kept after a small burn-in period should not be too surprising.

6.2 Applications to Gene expression Data

In this section we report the results of applying the Potts model clustering procedure to two different gene expression data sets: the subtypes of acute lymphoblastic leukemia data (Yeoh et al. 2002), and the yeast cell cycle data (Cho et al. 1998). We normalized the points to have norm one. Hence working with the Euclidean distance was equivalent to working with the correlations between the points. Using these distances, we constructed a 10-nearest-neighbor graph for each data set. The kernel chosen was a Gaussian kernel, i.e. \( k(x_i, x_j) \propto \exp\{-||x_i - x_j||^2/(2\sigma^2)\} \) whose bandwidth \( \sigma \) was estimated adaptively as explained in Section 4.1. The final clusters were forced
to have at least five points (smaller clusters were merged with the corresponding most correlated clusters).

For comparison purposes we ran both the non-adaptive (fixed bandwidth) and penalized versions of Potts model clustering, and the (non-parametric) dendrogram sharpening method (McKinney 1995, Stanberry, Nandy and Cordes 2003) on both data sets. We also applied model-based clustering (Banfield and Raftery 1993) to the yeast cycle data. The same values of the burn-in and partitions-kept parameters were used for all three versions of Potts model clustering. Similarly to the Potts model clustering procedure described in this paper, the dendrogram sharpening and model-based Gaussian clustering methods do not require prior assumption about the number or location of clusters. The reader is referred to the papers cited above for detailed descriptions of these methods.

The ALL data consist of oligonucleotide microarray gene expression levels of 12558 genes for each of 360 ALL patients. Yeoh et al. (2002) divided the patients into seven diagnostic groups corresponding to six known leukemia subtypes (T-ALL, E2A-PBX1, BCR-ABL, TEL-AML1, MLL rearrangement, and Hyperploid> 50 chromosomes), and one unknown type, labeled OTHER. The data were taken from the Kent Ridge Bio-Medical Data Set Repository, where they have been split into training and test sets. For our experiments we selected the training set comprising 215 patients.

In view of the simulation results described in Section 6.1, 1000 partitions were kept after a burn-in period of 300. Figure 4 shows the results of the adaptive-bandwidth Potts model clustering. The vertical line corresponds to the temperature \( T_0 = 0.67 \) chosen according to the procedure described in Section 5.1. A comparison of the corresponding clustering structure with that obtained in (Yeoh et al. 2002) is shown in Table 1. The rows of this matrix represent the seven subtypes of ALL leukemia assigned by Yeoh et al. (2002), while the columns represent the six estimated clusters from the adaptive-bandwidth Potts model clustering algorithm. Each cell \((i, j)\) of this matrix counts the number of points shared in the corresponding \(i−th\) true and \(j−th\) estimated clusters. The associated adjusted Rand index between these two partitions is 0.56. We can conclude that adaptive Potts model clustering produced clusters very similar to the

![Figure 4: The mean standard deviations \( \sqrt{\text{MVar}(L,G = 6,T)} \), \( L = 2,\ldots,6 \), as a function of temperature, resulting from the application of the adaptive-bandwidth Potts model clustering on the ALL data. The vertical line corresponds to \( T_0 = 0.67 \).](image)
Table 1: Partition matrix for the subtypes of acute lymphoblastic ALL data.

<table>
<thead>
<tr>
<th>True Clusters</th>
<th>Estimated Clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 2 3 4 5 6</td>
</tr>
<tr>
<td>BCR-ABL</td>
<td>0 0 0 9 0 0</td>
</tr>
<tr>
<td>E2A-PBX1</td>
<td>18 0 0 0 0 0</td>
</tr>
<tr>
<td>Hyperploid&gt;50</td>
<td>0 14 0 28 0 0</td>
</tr>
<tr>
<td>MLL rearrangement</td>
<td>3 0 9 2 0 0</td>
</tr>
<tr>
<td>OTHERS</td>
<td>5 0 2 39 0 6</td>
</tr>
<tr>
<td>T-ALL</td>
<td>0 0 1 0 27 0</td>
</tr>
<tr>
<td>TEL-AML1</td>
<td>0 0 0 0 0 52</td>
</tr>
</tbody>
</table>

seven subtypes of ALL leukemia assigned by Yeoh et al. (2002); except that Hyperploid>50 and OTHERS appear difficult to separate, and BCR-ABL appears difficult to be discriminated from Hyperploid>50 and OTHERS. An important consequence of the clustering is the clear separation between E2A-PBX1 and MLL rearrangement subtypes from the others, since patients with these types of leukemia do not respond well to conventional treatments.

Both the non-adaptive and penalized Potts model clustering algorithms were run on these data with constant bandwidth equal to the mean of the distances among the points in the K-nearest-neighbor graph. Both methods yielded similar adjusted Rand index of 0.43 and 0.44 with seven and six clusters, respectively. The resulting partitions were similar to the one obtained with the adaptive-bandwidth Potts model clustering. However, several patients with Hyperploid>50, OTHERS and TEL-AML1 shared the same cluster in the partition yielded by non-adaptive Potts model clustering, whereas several patients with subtypes Hyperploid>50 and MLL rearrangement shared the same cluster in the partition produced by penalized Potts model clustering. The adaptive-bandwidth penalized Potts model clustering yielded six clusters with an associated adjusted Rand index of 0.53. The main difference from the partition given by the adaptive-bandwidth version is that OTHERS was split among five clusters. The dendrogram sharpening method yielded a three-cluster partition with an adjusted Rand index of 0.23.

The yeast cell cycle data record the fluctuations of the expression levels of about 6000 genes over two cell cycles comprising 17 time points. We use the 5-phase subset of the data (Cho et al. 1998). It consists of 420 genes of which 386 have been assigned to one of a five phases of the cells cycle. The clustering results should reveal five groups of genes associated with the five phases. We ran the adaptive-bandwidth Potts model clustering with 500 iterations of the SWA: 250 partitions were kept after a burn-in period of 250. Despite the small number of partitions kept in the analysis, Potts model clustering was able to find a good nine-cluster partition of the data. Table 2 shows the corresponding partition matrix. The associated adjusted Rand index was slightly over 0.46. A run of the algorithm at the same temperature that gave rise to the nine-cluster partition, but forcing the cluster size to be least 20, yielded a six-cluster partition with an associated adjusted Rand index of 0.45. Hence, it is important to have a good prior estimate of the size of the smallest cluster. The non-adaptive Potts model clustering algorithm yielded 16 clusters with an associated adjusted Rand index of 0.43. Penalized Potts model clustering yielded nine clusters with an associated adjusted Rand index of 0.45. Both the adaptive and penalized algorithms yielded similar clustering structures for this data. Yeung et al. (2001) analyzed the labeled subset of these data using model-based clustering based on Gaussian mixtures (Banfield and Raftery 1993). They reported four clusters with an adjusted Rand index of about 0.43. The
Table 2: Partition matrix for the yeast cell cycle data.

<table>
<thead>
<tr>
<th>True Clusters</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>51</td>
<td>5</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td>0</td>
<td>117</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>0</td>
<td>34</td>
<td>13</td>
<td>10</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>17</td>
<td>10</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>44</td>
<td></td>
</tr>
</tbody>
</table>

dendrogram sharpening method yielded a four-cluster partition with an adjusted Rand index of 0.45.

6.3 Exploring Kernel Density Estimation

In this section we explore the connection between Potts model clustering and kernel density estimation as outlined in Section 4.3. We compare the three different strategies of bandwidth estimation described in Sections 4.1 and 4.2 on two one-dimensional and one two-dimensional data sets: an artificial data set, the galaxy data, and the Old Faithful data.

We ran Potts model clustering with bandwidth estimation using the Euclidean distance as a measure of similarity between the points, and the Gaussian kernel. We ran the modified Swendsen-Wang Metropolis-Hastings algorithms (see Sections 4.1 and 4.2) with 600 complete iterations. The first 300 iterations were discarded from the analysis. The remainder 300 iterations were used to cluster the data and to compute the cluster size variance in the largest six clusters. The final clusters were forced to have at least five points. Unless otherwise noted, all runs were initialized with constant bandwidth equal to the mean distance between any two points. We observed that all three bandwidth estimation algorithms (Gamma bandwidth selection, and Gaussian and Laplace smoothing priors) selected clusterings at similar temperatures for these data sets. Hence the bandwidth estimation did not appear to influence much the choice of temperature for clustering, and the clustering themselves. In other words, any of the bandwidth estimation algorithms would do equally well for clustering purposes. The main differences were in the bandwidth estimates.

The artificial data set was created with the purpose of comparing the kernel density estimates with the true density of the data. These data were generated from a Gaussian mixture with five components centered at \(-0.3, 0.0, 0.2, 0.8, \text{ and } 1.0\); and with variances equal to \(1.0, 2.0, 0.7, 0.4, \text{ and } 0.2\), respectively. The associated component proportions were equal to (up to a normalizing constant) \(0.2, 0.1, 0.1, 0.2, \text{ and } 0.3\). 150 points were drawn at random from this distribution. Figure 5 shows the bandwidths means for the Gamma updates (top left), the Gaussian updates with \(\sigma^2 = 0.1\), \(\sigma^2 = 1.0\) (top right), and the Laplace updates with prior parameter \(\Delta = 100\) (bottom left) and \(\Delta = 10000\) (bottom right). The associated kernel density estimators as well as the true density (top left) and the adaptive bandwidth kernel density estimator (top right) are depicted in Figure 6. One can clearly appreciate the smoothness in the bandwidths introduced by the Gaussian smoothing prior, and the nearly piece-wise constant shape yielded by the Laplace smoothing prior. All associated kernel density estimators look very similar to each other. They all introduce smoothness to the estimator as compared with the one based only on the adaptive bandwidth.
Figure 5: Artificial dataset: Bandwidths means for different Metropolis-Hastings bandwidth updating strategies: Gamma (top left), Gaussian with $\sigma^2 = 0.1, \sigma^2 = 1.0$ (top right), and Laplace with $\Delta = 100$ (bottom left) and $\Delta = 10000$ (bottom right).

The galaxy dataset as provided with S-Plus version 6.2 consists of 323 measurements of the radial velocity (in km/second) of a spiral galaxy (NGC7531) measured at points in the area of the sky covered by it (Buta 1987, Chambers and Hastie 1992). Figure 7 shows the bandwidth medians and the associated kernel density estimators yielded by Potts model clustering with Gamma and Gaussian penalty ($\sigma^2 = 1.0$) updates. The bandwidth smoothness introduced by the Gaussian smoothing prior is obvious. The figures clearly show eight to nine modes in the density estimates which correspond to the clusters found by the Potts model algorithms.

The Old Faithful dataset as provided with the S-Plus version 6.2 consists of 299 measurements of the waiting time between eruptions and the duration of the eruption for the Old Faithful geyser in Yellowstone National Park (Azzalini and Bowman 1990). The bandwidth medians yielded by the different bandwidth update algorithms are shown in Figure 8. The corresponding two-dimensional contours and kernel density estimates are shown in Figure 9. Note the spikes in the bandwidths near previous durations 2.0 and 4.0. These mark the boundaries between the two main clusters of points. Also note that the bandwidths tend to increase in the “empty” region. The spikes are probably due to the sharp increase in the density at the clusters. The Laplace smoothing prior updates yield almost piece-wise constant bandwidths within the two main clusters and the empty region. The updates are highly variable in the boundary regions. The contours in Figure 9 show the smoothness introduced by the Gaussian and Laplace smoothing priors. Overall, the Laplace updates appeared to have produced the smoothest looking kernel density estimator, although the Gaussian updates seemed to have yielded the smoothest bandwidths.
Figure 6: Artificial dataset: Kernel density estimator means for different Metropolis-Hastings bandwidth updates strategies: True density (top left), kernel density with adaptive bandwidth (top right), Gamma (center left), Gaussian with $\sigma^2 = 0.1$, $\sigma^2 = 1.0$ (center right), and Laplace with $\Delta = 100$ (bottom left) and $\Delta = 10000$ (bottom right).

7 Discussion

The main contribution of this paper is to uncover and exploit the close connection between Potts model clustering and kernel K-means and kernel density estimation. Interpreting the Hamiltonian of the Potts model in terms of the conditional densities given the cluster labels motivates a variant of Potts model that incorporates a penalty for unequal cluster sizes. A modification of the Wolff algorithm allows us to simulate configurations from the distribution defined by this penalized Hamiltonian, leading to penalized Potts model clustering. The link to kernel density estimation suggests replacing constant bandwidth with adaptive bandwidth kernels, a generalization long recognized as advantageous in the context of density estimation that also turns out to be beneficial for clustering.

There are several directions for future work:

(i) More general penalty terms for penalized Potts model clustering. One could, for example, use a different kernel $k_p(x_i, x_j)$ for interactions between non-neighbor points. Note that the algorithm outlined in this paper uses a constant penalty ($k_p(x_i, x_j) = 1$) for non-neighbor points with the same label.

(ii) More computationally efficient ways of choosing the temperature. Our current method requires simulating configurations at several different temperatures. It would be more efficient if a good temperature could be discovered in a single run of the algorithm. We think a strategy close in spirit to simulated tempering (Marinari and Parisi 1992, Geyer and Thompson 2009)
Figure 7: Galaxy dataset: Bandwidths medians and associated kernel density estimators for different Metropolis-Hastings bandwidth updates strategies: Gamma (left column), and Gaussian with $\sigma_0^2 = \sigma^2 = 1.0$ (right column).

...1995) and parallel tempering (Geyer 1991) may be worth investigating.

(iii) Extension to semi-supervised learning. In semi-supervised learning one is given the true labels for a (typically small) subset of the observations. This information could be incorporated by assigning a large similarity to pairs of observations known to have the same label, and a small similarity to pairs known to have different labels.

References


Figure 8: Old Faithful: Bandwidth medians yielded by the Gamma update (left), Gaussian with $\sigma_0^2 = 0.01$, $\sigma^2 = 1.0$ (center), and Laplace with $\Delta = 100$ (right). The horizontal axes correspond to the waiting time to eruption (upper plots) and to the previous eruption duration (lower plots).


Figure 9: Old Faithful: Kernel density estimators associated with the bandwidth medians yielded by the Gamma update (left), Gaussian with $\sigma_0^2 = 0.01$, $\sigma^2 = 1.0$ (center), and Laplace with $\Delta = 100$ (right).


The normalized cut between any two clusters \( k \) and \( k' \) is defined as (Shi and Malik 2000, Meila and Xu 2003, Yu and Shi 2003)

\[
NCut(k, k') = \left( \frac{1}{\text{vol}(k)} + \frac{1}{\text{vol}(k')} \right) \sum_{i=1}^{n} \sum_{j=1}^{n} z_{ki}z_{kj}k_{ij}
\]  

(26)
where \( \text{vol}(\ell) = \sum_{i=1}^{n} \sum_{j=1}^{n} z_{ij} k_{ij} \), \( \ell = 1, \ldots, q \). The MNCut of any given partition is then defined as

\[
\text{MNCut} = \sum_{k=1}^{q} \sum_{k'=k+1}^{q} \text{NCut}(k, k') = \frac{1}{2} \sum_{k=1}^{q} \sum_{k'=1}^{q} \text{NCut}(k, k') - \frac{1}{2} \sum_{k=1}^{q} \text{NCut}(k, k) \tag{27}
\]

The goal of MNCut is to find the set of labels \( \{z_{ki}\} \) that minimize (27). Using (26), one easily obtains

\[
\text{MNCut} = \frac{1}{2} \sum_{k=1}^{q} \sum_{k'=1}^{q} \left[ \sum_{i=1}^{n} \sum_{j=1}^{n} z_{kj} z_{k'i} k_{ij} \right] \left( \left[ \sum_{i=1}^{n} \sum_{j=1}^{n} z_{ki} k_{ij} \right]^{-1} + \left[ \sum_{i=1}^{n} \sum_{j=1}^{n} z_{k'i} k_{ij} \right]^{-1} \right) \\
- \frac{1}{2} \sum_{k=1}^{q} \sum_{k'=1}^{q} \sum_{i=1}^{n} \sum_{j=1}^{n} z_{kj} z_{k'i} k_{ij} \times 2 \left[ \sum_{i=1}^{n} \sum_{j=1}^{n} z_{ki} k_{ij} \right]^{-1} \\
= \sum_{k=1}^{q} \sum_{k'=1}^{q} \sum_{i=1}^{n} \sum_{j=1}^{n} z_{kj} z_{k'i} k_{ij} \sum_{i=1}^{n} \sum_{j=1}^{n} z_{ki} k_{ij} - \sum_{k=1}^{q} \sum_{k'=1}^{q} \sum_{i=1}^{n} \sum_{j=1}^{n} z_{kj} z_{k'i} k_{ij} \\
= q \sum_{k=1}^{q} \sum_{i=1}^{n} \sum_{j=1}^{n} z_{kj} z_{k'i} k_{ij} - q \sum_{k=1}^{q} \sum_{i=1}^{n} \sum_{j=1}^{n} z_{kj} z_{k'i} k_{ij} \\
= q - n \sum_{k=1}^{q} \sum_{i=1}^{n} \sum_{j=1}^{n} z_{kj} z_{k'i} k_{ij} = q - \sum_{k=1}^{q} w_k \gamma_k R_k
\]

where \( w_k = n / \sum_{j=1}^{n} \hat{p}(x_j | k) \). Writing \( \sum_{k=1}^{q} w_k \gamma_k R_k \) as

\[
\sum_{k=1}^{q} w_k \gamma_k R_k = \sum_{k=1}^{q} \frac{\text{mass assigned to cluster } k \text{ given cluster } k}{\text{total mass in cluster } k \text{ given cluster } k}
\]

one sees that MNCut tries to maximize the total mass assigned to the clusters so that the data points assigned to the corresponding clusters account for almost all the mass associated with the conditional densities.

A straightforward calculation shows that, as in the kernel \( K \)-means and Potts model clustering cases, \( \sum_{k=1}^{q} w_k \gamma_k R_k \) is another way to write (2) with weights given by

\[
w(i, j, \{z_{ki}\}, k) = \begin{cases} 
0 & \text{if } \delta_{ij} = 0 \\
\left( \sum_{i=1}^{n} \sum_{j=1}^{n} k(x_i, x_j) \right)^{-1} & \text{if } z_{ki} = z_{kj} = 1
\end{cases}
\]

Hence minimizing (27) is again linked to some sort of weighted \( K \)-means procedure with cluster dependent weights \( w_k \). Indeed, it is straightforward to verify that a weighted \( K \)-means with weights \( w_k \) maximizes

\[
\sum_{k=1}^{q} w_k \gamma_k (R_k - 1).
\]

Note that \( \sum_{k=1}^{q} w_k \gamma_k = \sum_{k=1}^{q} \gamma_k R_k / m(k) \), where \( m(k) = \sum_{j=1}^{n} \hat{p}(x_j | k) / n_k \) is the “average” mass in cluster \( k \). The weighted \( K \)-means with weights given by \( w_k \)'s penalizes clusterings with large variations in average masses across the clusters. Thus, unlike the weighted \( K \)-means with weights \( \gamma_k \)'s that penalizes unequal cluster sizes, the MNCut induced weighted \( K \)-means penalizes unequal cluster masses.