Journal of Machine Learning Research 6 (2005) 1345–1382 Submitted 9/04; Revised 4/05; Published 9/05

Clustering on the Unit Hypersphere using von Mises-Fisher Distributions

Arindam Banerjee ABANERJE@ECE.UTEXAS.EDU
Department of Electrical and Computer Engineering
University of Texas at Austin
Austin, TX 78712, USA

Inderjit S. Dhillon INDERJIT@CS.UTEXAS.EDU
Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712, USA

Joydeep Ghosh GHOSH@ECE.UTEXAS.EDU
Department of Electrical and Computer Engineering
University of Texas at Austin
Austin, TX 78712, USA

Suvrit Sra SUVRIT@CS.UTEXAS.EDU
Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712, USA

Editor: Greg Ridgeway

Abstract

Several large scale data mining applications, such as text categorization and gene expression analysis, involve high-dimensional data that is also inherently directional in nature. Often such data is L_2 normalized so that it lies on the surface of a unit hypersphere. Popular models such as (mixtures of) multi-variate Gaussians are inadequate for characterizing such data. This paper proposes a generative mixture-model approach to clustering directional data based on the von Mises-Fisher (vMF) distribution, which arises naturally for data distributed on the unit hypersphere. In particular, we derive and analyze two variants of the Expectation Maximization (EM) framework for estimating the mean and concentration parameters of this mixture. Numerical estimation of the concentration parameters is non-trivial in high dimensions since it involves functional inversion of ratios of Bessel functions. We also formulate two clustering algorithms corresponding to the variants of EM that we derive. Our approach provides a theoretical basis for the use of cosine similarity that has been widely employed by the information retrieval community, and obtains the spherical kmeans algorithm (kmeans with cosine similarity) as a special case of both variants. Empirical results on clustering of high-dimensional text and gene-expression data based on a mixture of vMF distributions show that the ability to estimate the concentration parameter for each vMF component, which is not present in existing approaches, yields superior results, especially for difficult clustering tasks in high-dimensional spaces.

Keywords: clustering, directional distributions, mixtures, von Mises-Fisher, expectation maximization, maximum likelihood, high dimensional data

1. Introduction

Clustering or segmentation of data is a fundamental data analysis step that has been actively investigated by many research communities over the past few decades (Jain and Dubes, 1988). However, traditional methods for clustering data are severely challenged by a variety of complex characteristics exhibited by certain recent data sets examined by the machine learning and data mining communities. These data sets, acquired from scientific domains and the world wide web, also impose significant demands on scalability, visualization and evaluation of clustering methods (Ghosh, 2003). In this paper we focus on clustering objects such as text documents and gene expressions, where the complexity arises from their representation as vectors that are not only very high dimensional (and often sparse) but also *directional*, i.e., the vector direction is relevant, not its magnitude.

One can broadly categorize clustering approaches to be either generative (also known as parametric or probabilistic) (Smyth, 1997; Jaakkola and Haussler, 1999) or discriminative (non-parametric) (Indyk, 1999; Schölkopf and Smola, 2001). The performance of an approach, and of a specific method within that approach, is quite data dependent; there is no clustering method that works the best across all types of data. Generative models, however, often provide greater insight into the anatomy of the clusters. A lot of domain knowledge can be incorporated into generative models, so that clustering of data uncovers specific desirable patterns that one is looking for.

Clustering algorithms using the generative model framework, often involve an appropriate application of the Expectation Maximization (EM) algorithm (Dempster et al., 1977; McLachlan and Krishnan, 1997) on a properly chosen statistical generative model for the data under consideration. For vector data, there are well studied clustering algorithms for popular generative models such as a mixture of multivariate Gaussians, whose effect is analogous to the use of Euclidean or Mahalanobis type distances as the chosen measure of distortion from the discriminative perspective.

The choice of a particular distortion measure (or the corresponding generative model) can be crucial to the performance of a clustering procedure. There are several domains where methods based on minimizing Euclidean distortions yield poor results (Strehl et al., 2000). For example, studies in information retrieval applications convincingly demonstrate *cosine similarity* to be a more effective measure of similarity for analyzing and clustering text documents. In this domain, there is substantial empirical evidence that normalizing the data vectors helps to remove the biases induced by the length of a document and provide superior results (Salton and McGill, 1983; Salton and Buckley, 1988). Further, the spherical kmeans (`spkmeans`) algorithm (Dhillon and Modha, 2001), that performs kmeans using cosine similarity instead of Euclidean distortion, has been found to work well for text clustering. Data Sets from such domains, where similarity measures such as cosine, Jaccard or Dice (Rasmussen, 1992) are more effective than measures derived from Mahalanobis type distances, possess intrinsic "directional" characteristics, and are hence better modeled as *directional data* (Mardia and Jupp, 2000).[1]

There are many other important domains such as bioinformatics (e.g., Eisen et al. (1998)), collaborative filtering (e.g., Sarwar et al. (2001)) etc., in which directional data is encountered. Consider the Pearson correlation coefficient, which is a popular similarity measure in both these domains . Given $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$, the Pearson product moment correlation between them is given by $\rho(\boldsymbol{x}, \boldsymbol{y}) = \frac{\sum_{i=1}^d (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^d (x_i - \bar{x})^2}\sqrt{\sum_{i=1}^d (y_i - \bar{y})^2}}$, where $\bar{x} = \frac{1}{d}\sum_{i=1}^d x_i$, $\bar{y} = \frac{1}{d}\sum_{i=1}^d y_i$. Consider the mapping $\boldsymbol{x} \mapsto \tilde{\boldsymbol{x}}$ such that $\tilde{x}_i = \frac{x_i - \bar{x}}{\sqrt{\sum_{i=1}^d (x_i - \bar{x})^2}}$, and a similar mapping for $\boldsymbol{y}$. Then we have $\rho(\boldsymbol{x}, \boldsymbol{y}) = \tilde{\boldsymbol{x}}^T \tilde{\boldsymbol{y}}$. Moreover,

1. This paper treats L_2 normalized data and directional data as synonymous.

$\|\tilde{\boldsymbol{x}}\|_2 = \|\tilde{\boldsymbol{y}}\|_2 = 1$. Thus, the Pearson correlation is exactly the cosine similarity between $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{y}}$. Hence, analysis and clustering of data using Pearson correlations is essentially a clustering problem for directional data.

1.1 Contributions

In this paper[2] we present a generative model, consisting of a mixture of von Mises-Fisher (vMF) distributions, tailored for directional data distributed on the surface of a unit hypersphere. We derive two clustering algorithms based on EM for estimating the parameters of the mixture model from first principles. The algorithm involves estimating a *concentration* parameter, κ, for high dimensional data. The ability to adapt κ on a per-component basis leads to substantial performance improvements over existing generative approaches to modeling directional data. We show a connection between the proposed methods and a class of existing algorithms for clustering high-dimensional directional data. In particular, our generative model has the same relation to `spkmeans` as a model based on a mixture of unit covariance Gaussians has to classical `kmeans` that uses squared Euclidean distances. We also present detailed experimental comparisons of the proposed algorithms with `spkmeans` and one of its variants. Our formulation uncovers the theoretical justification behind the use of the cosine similarity measure that has largely been ad-hoc, i.e., based on empirical or intuitive justification, so far.

Other key contributions of the paper are:

- It exposes the vMF model to the learning community and presents a detailed parameter estimation method for learning mixtures of vMF distributions in high-dimensions. Previously known parameter estimates for vMF distributions are reasonable only for low-dimensional data (typically only 2 or 3 dimensional data is considered) and are hence not applicable to many modern applications such as text clustering.

- We show that hard assignments maximize a tight lower bound on the incomplete log-likelihood function. In addition, our analysis of hard assignments is applicable to any mixture model learning using EM. This result is particularly important when using mixtures of vMFs since the computational needs for hard assignments are lower than what is required for the standard soft assignments (E-step) for these models.

- Extensive experimental results are provided on benchmark text and gene-expression data sets to show the efficacy of the proposed algorithms for high-dimensional, directional data. Good results are obtained even for fairly skewed data sets. A recent study (Banerjee and Langford, 2004) using PAC-MDL bounds for evaluation of clustering algorithms also demonstrated the efficacy of the proposed approaches.

- An explanation of the superior performance of the soft-assignment algorithm is obtained by drawing an analogy between the observed cluster formation behavior and locally adaptive annealing. See Section 7 for further details.

2. An earlier, short version of this paper appeared as: *Generative Model-based Clustering of Directional Data* in Proceedings KDD, 2003.

1.2 Related Work

There has been an enormous amount of work on clustering a wide variety of data sets across multiple disciplines over the past fifty years (Jain and Dubes, 1988). The methods presented in this paper are tailored for high-dimensional data with directional characteristics, rather than for arbitrary data sets. In the learning community, perhaps the most widely studied high-dimensional directional data stem from text documents represented by vector space models. Much of the work in this domain uses discriminative approaches (Steinbach et al., 2000; Zhao and Karypis, 2004). For example, hierarchical agglomerative methods based on cosine, Jaccard or Dice coefficients were dominant for text clustering till the mid-1990s (Rasmussen, 1992). Over the past few years several new approaches, ranging from spectral partitioning (Kannan et al., 2000; Zhao and Karypis, 2004), to the use of generative models from the exponential family, e.g., mixture of multinomials or Bernoulli distributions (Nigam et al., 2000) etc., have emerged. A fairly extensive list of references on generative approaches to text clustering can be found in (Zhong and Ghosh, 2003a).

Of particular relevance to this work is the `spkmeans` algorithm (Dhillon and Modha, 2001), which adapts the `kmeans` algorithm to normalized data by using the cosine similarity for cluster allocation, and also by re-normalizing the cluster means to unit length. The `spkmeans` algorithm is superior to regular `kmeans` for high-dimensional text data, and competitive or superior in both performance and speed to a wide range of other existing alternatives for text clustering (Strehl et al., 2000). It also provides better characterization of clusters in terms of their top representative or discriminative terms.

The larger topic of clustering very high-dimensional data (dimension in the thousands or more), irrespective of whether it is directional or not, has also attracted great interest lately. Again, most of the proposed methods of dealing with the curse of dimensionality in this context follow a density-based heuristic or a discriminatory approach (Ghosh, 2003). Among generative approaches for clustering high-dimensional data, perhaps the most noteworthy is one that uses low dimensional projections of mixtures of Gaussians (Dasgupta, 1999). It turns out that one of our proposed methods alleviates problems associated with high dimensionality via an implicit local annealing behavior.

The vMF distribution is known in the literature on directional statistics (Mardia and Jupp, 2000), and the maximum likelihood estimates (MLE) of the parameters have been given for a single distribution. Recently Piater (2001) obtained parameter estimates for a mixture for circular, i.e., 2-dimensional vMFs. In an Appendix to his thesis, Piater (2001) starts on an EM formulation for 2-D vMFs but cites the difficulty of parameter estimation (especially κ) and eventually avoids doing EM in favor of another numerical gradient descent based scheme. Mooney et al. (2003) use a mixture of two circular von Mises distributions to estimate the parameters using a quasi-Newton procedure. Wallace and Dowe (2000) perform mixture modeling for circular von Mises distributions and have produced a software called Snob that implements their ideas. McLachlan and Peel (2000) discuss mixture analysis of directional data and mention the possibility of using Fisher distributions (3-dimensional vMFs), but instead use 3-dimensional Kent distributions (Mardia and Jupp, 2000). They also mention work related to the clustering of directional data, but all the efforts included by them are restricted to 2-D or 3-D vMFs. Indeed, McLachlan and Peel (2000) also draw attention to the difficulty of parameter estimation even for 3-D vMFs. Even for a single component, the maximum-likelihood estimate for the concentration parameter κ involves inverting a ratio of two Bessel functions, and current ways of approximating this operation are inadequate for high-dimensional data. It turns out that our estimate for κ translates into a substantial improvement in the empirical results.

The connection between a generative model involving vMF distributions with constant κ and the `spkmeans` algorithm was first observed by Banerjee and Ghosh (2002). A variant that could adapt in an on-line fashion leading to balanced clustering solutions was developed by Banerjee and Ghosh (2004). Balancing was encouraged by taking a frequency-sensitive competitive learning approach in which the concentration of a mixture component was made inversely proportional to the number of data points already allocated to it. Another online competitive learning scheme using vMF distributions for minimizing a KL-divergence based distortion was proposed by Sinkkonen and Kaski (2001). Note that the full EM solution was not obtained or employed in either of these works. Recently a detailed empirical study of several generative models for document clustering, including a simple mixture-of-vMFs model that constrains the concentration κ to be the same for all mixture components during any iteration was presented by Zhong and Ghosh (2003b). Even with this restriction, this model was superior to both hard and soft versions of multivariate Bernoulli and multinomial models. These positive results further motivate the current paper in which we present the general EM solution for parameter estimation of a mixture of vMF distributions. This enhancement leads to even better clustering performance for difficult clustering tasks: when clusters overlap, when cluster sizes are skewed, and when cluster sizes are small relative to the dimensionality of the data. In the process, several new, key insights into the nature of hard vs. soft mixture modeling and the behavior of vMF based mixture models are obtained.

The remainder of the paper is organized as follows. We review the multi-variate vMF distribution in Section 2. In Section 3 we introduce a generative model using a mixture of vMF distributions. We then derive the maximum likelihood parameter estimates of this model by employing an EM framework. Section 4 highlights our new method of approximating κ and also presents a mathematical analysis of hard assignments. Sections 3 and 4 form the basis for two clustering algorithms using soft and hard-assignments respectively, that are proposed in Section 5. Detailed experimental results and comparisons with other algorithms are offered in Section 6. A discussion on the behavior of our algorithms and a connection with simulated annealing follows in Section 7. Section 8 concludes our paper and highlights some possible directions for future work.

Notation. Bold faced variables, e.g., $\boldsymbol{x}, \boldsymbol{\mu}$ represent vectors; the norm $\|\cdot\|$ denotes the L_2 norm; sets are represented by script-style upper-case letters, e.g., $\mathcal{X}$, $\mathcal{Z}$. The set of reals is denoted by $\mathbb{R}$, while $\mathbb{S}^{d-1}$ denotes the $(d-1)$-dimensional sphere embedded in $\mathbb{R}^d$. Probability density functions are denoted by lower case letters such as f, p, q and the probability of a set of events is denoted by P. If a random vector $\boldsymbol{x}$ is distributed as $p(\cdot)$, expectations of functions of $\boldsymbol{x}$ are denoted by $E_p[\cdot]$.

2. Preliminaries

In this section, we review the von Mises-Fisher distribution and maximum likelihood estimation of its parameters from independent samples.

2.1 The von Mises-Fisher (vMF) Distribution

A d-dimensional unit random vector $\boldsymbol{x}$ (i.e., $\boldsymbol{x} \in \mathbb{R}^d$ and $\|\boldsymbol{x}\| = 1$, or equivalently $\boldsymbol{x} \in \mathbb{S}^{d-1}$) is said to have d-variate von Mises-Fisher (vMF) distribution if its probability density function is given by

$$f(\boldsymbol{x}|\boldsymbol{\mu}, \kappa) = c_d(\kappa) e^{\kappa \boldsymbol{\mu}^T \boldsymbol{x}}, \tag{2.1}$$

where $\|\boldsymbol{\mu}\| = 1$, $\kappa \geq 0$ and $d \geq 2$. The normalizing constant $c_d(\kappa)$ is given by

$$c_d(\kappa) = \frac{\kappa^{d/2-1}}{(2\pi)^{d/2} I_{d/2-1}(\kappa)}, \tag{2.2}$$

where $I_r(\cdot)$ represents the modified Bessel function of the first kind and order r. The density $f(\boldsymbol{x}|\boldsymbol{\mu},\kappa)$ is parameterized by the mean direction $\boldsymbol{\mu}$, and the *concentration* parameter κ, so-called because it characterizes how strongly the unit vectors drawn according to $f(\boldsymbol{x}|\boldsymbol{\mu},\kappa)$ are concentrated about the mean direction $\boldsymbol{\mu}$. Larger values of κ imply stronger concentration about the mean direction. In particular when $\kappa = 0$, $f(\boldsymbol{x}|\boldsymbol{\mu},\kappa)$ reduces to the uniform density on $\mathbb{S}^{d-1}$, and as $\kappa \to \infty$, $f(\boldsymbol{x}|\boldsymbol{\mu},\kappa)$ tends to a point density. The interested reader is referred to Mardia and Jupp (2000), Fisher (1996) or Dhillon and Sra (2003) for details on vMF distributions.

The vMF distribution is one of the simplest parametric distributions for directional data, and has properties analogous to those of the multi-variate Gaussian distribution for data in $\mathbb{R}^d$. For example, the maximum entropy density on $\mathbb{S}^{d-1}$ subject to the constraint that $E[\boldsymbol{x}]$ is fixed is a vMF density (see Rao (1973, pp. 172–174) and Mardia (1975) for details).

2.2 Maximum Likelihood Estimates

In this section we look briefly at maximum likelihood estimates for the parameters of a single vMF distribution. The detailed derivation can be found in Appendix A. Let $\mathcal{X}$ be a finite set of sample unit vectors drawn independently following $f(\boldsymbol{x}|\boldsymbol{\mu},\kappa)$ (2.1), i.e.,

$$\mathcal{X} = \{\boldsymbol{x}_i \in \mathbb{S}^{d-1} \mid \boldsymbol{x}_i \text{ drawn following } f(\boldsymbol{x}|\boldsymbol{\mu},\kappa) \text{ for } 1 \leq i \leq n\}.$$

Given $\mathcal{X}$ we want to find maximum likelihood estimates for the parameters $\boldsymbol{\mu}$ and κ of the distribution $f(\boldsymbol{x}|\boldsymbol{\mu},\kappa)$. Assuming the $\boldsymbol{x}_i$ to be independent and identically distributed, we can write the log-likelihood of $\mathcal{X}$ as

$$\ln P(\mathcal{X}|\boldsymbol{\mu},\kappa) = n \ln c_d(\kappa) + \kappa \boldsymbol{\mu}^T \boldsymbol{r}, \tag{2.3}$$

where $\boldsymbol{r} = \sum_i \boldsymbol{x}_i$. To obtain the maximum likelihood estimates of $\boldsymbol{\mu}$ and κ, we have to maximize (2.3) subject to the constraints $\boldsymbol{\mu}^T\boldsymbol{\mu} = 1$ and $\kappa \geq 0$. After some algebra (details may be found in Section A.1) we find that the MLE solutions $\hat{\boldsymbol{\mu}}$ and $\hat{\kappa}$ may be obtained from the following equations:

$$\hat{\boldsymbol{\mu}} = \frac{\boldsymbol{r}}{\|\boldsymbol{r}\|} = \frac{\sum_{i=1}^n \boldsymbol{x}_i}{\|\sum_{i=1}^n \boldsymbol{x}_i\|}, \tag{2.4}$$

$$\text{and} \qquad \frac{I_{d/2}(\hat{\kappa})}{I_{d/2-1}(\hat{\kappa})} = \frac{\|\boldsymbol{r}\|}{n} = \bar{r}. \tag{2.5}$$

Since computing $\hat{\kappa}$ involves an implicit equation (2.5) that is a ratio of Bessel functions, it is not possible to obtain an analytic solution, and we have to take recourse to numerical or asymptotic methods to obtain an approximation (see Section 4.1).

3. EM on a Mixture of vMFs (moVMF)

We now consider a mixture of k vMF (moVMF) distributions that serves as a generative model for directional data. Subsequently we derive the update equations for estimating the mixture-density parameters from a given data set using the Expectation Maximization (EM) framework. Let $f_h(\boldsymbol{x}|\theta_h)$

denote a vMF distribution with parameter $\theta_h = (\boldsymbol{\mu}_h, \kappa_h)$ for $1 \le h \le k$. Then a mixture of these k vMF distributions has a density given by

$$f(\boldsymbol{x}|\Theta) = \sum_{h=1}^{k} \alpha_h f_h(\boldsymbol{x}|\theta_h), \tag{3.1}$$

where $\Theta = \{\alpha_1, \cdots, \alpha_k, \theta_1, \cdots, \theta_k\}$ and the α_h are non-negative and sum to one. To sample a point from this mixture density we choose the h-th vMF randomly with probability α_h, and then sample a point (on $\mathbb{S}^{d-1}$) following $f_h(\boldsymbol{x}|\theta_h)$. Let $\mathcal{X} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n\}$ be a data set of n independently sampled points that follow (3.1). Let $\mathcal{Z} = \{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_n\}$ be the corresponding set of hidden random variables that indicate the particular vMF distribution from which the points are sampled. In particular, $\boldsymbol{z}_i = h$ if $\boldsymbol{x}_i$ is sampled from $f_h(\boldsymbol{x}|\theta_h)$. Assuming that the values in the set $\mathcal{Z}$ are known, the log-likelihood of the observed data is given by

$$\ln P(\mathcal{X}, \mathcal{Z}|\Theta) = \sum_{i=1}^{n} \ln\left(\alpha_{\boldsymbol{z}_i} f_{\boldsymbol{z}_i}(\boldsymbol{x}_i|\theta_{\boldsymbol{z}_i})\right). \tag{3.2}$$

Obtaining maximum likelihood estimates for the parameters would have been easy were the $\boldsymbol{z}_i$ truly known. Unfortunately that is not the case, and (3.2) is really a random variable dependent on the distribution of $\mathcal{Z}$—this random variable is usually called the *complete data log-likelihood*. For a given $(\mathcal{X}, \Theta)$, it is possible to estimate the most likely conditional distribution of $\mathcal{Z}|(\mathcal{X}, \Theta)$, and this estimation forms the E-step in an EM framework.

Using an EM approach for maximizing the expectation of (3.2) with the constraints $\boldsymbol{\mu}_h^T \boldsymbol{\mu}_h = 1$ and $\kappa_h \ge 0$, we obtain (see Appendix A.2),

$$\alpha_h = \frac{1}{n} \sum_{i=1}^{n} p(h|\boldsymbol{x}_i, \Theta), \tag{3.3}$$

$$\boldsymbol{r}_h = \sum_{i=1}^{n} \boldsymbol{x}_i p(h|\boldsymbol{x}_i, \Theta), \tag{3.4}$$

$$\hat{\boldsymbol{\mu}}_h = \frac{\boldsymbol{r}_h}{\|\boldsymbol{r}_h\|}, \tag{3.5}$$

$$\frac{I_{d/2}(\hat{\kappa}_h)}{I_{d/2-1}(\hat{\kappa}_h)} = \frac{\|\boldsymbol{r}_h\|}{\sum_{i=1}^{n} p(h|\boldsymbol{x}_i, \Theta)}. \tag{3.6}$$

Observe that (3.5) and (3.6) are intuitive generalizations of (2.4) and (2.5) respectively, and they correspond to an M-step in an EM framework. Given these parameter updates, we now look at schemes for updating the distributions of $\mathcal{Z}|(\mathcal{X}, \Theta)$ (i.e., an E-step) to maximize the likelihood of the data given the parameters estimates above.

From the standard EM framework, the distribution of the hidden variables (Neal and Hinton, 1998; Bilmes, 1997) is given by

$$p(h|\boldsymbol{x}_i, \Theta) = \frac{\alpha_h\, f_h(\boldsymbol{x}_i|\Theta)}{\sum_{l=1}^{k} \alpha_l\, f_l(\boldsymbol{x}_i|\Theta)}. \tag{3.7}$$

It can be shown (Collins, 1997) that the *incomplete data log-likelihood*, $\ln p(\mathcal{X}|\Theta)$, is non-decreasing at each iteration of the parameter and distribution updates. Iteration over these two updates provides the foundation for our `soft-moVMF` algorithm given in Section 5.

Our second update scheme is based on the widely used hard-assignment heuristic for unsupervised learning. In this case, the distribution of the hidden variables is given by

$$q(h|\boldsymbol{x}_i,\Theta) = \begin{cases} 1, & \text{if } h = \underset{h'}{\operatorname{argmax}} \; p(h'|\boldsymbol{x}_i,\Theta), \\ 0, & \text{otherwise.} \end{cases} \tag{3.8}$$

We analyze the above hard-assignment rule in Section 4, and show that it maximizes a lower bound on the incomplete data log-likelihood. Iteration over the M-step and the hard-assignment rule leads to the `hard-moVMF` algorithm given in Section 5.

4. Handling Large and High-Dimensional Data Sets

Although the mixture model outlined in section 3 seems quite straight-forward, there are some of critical issues that need to be addressed before one can apply the model to large high-dimensional data sets:

A. How to compute $\kappa_h, h = 1,\dots,k$ from (3.6) for high-dimensional data?

B. Is it possible to significantly reduce computations and still get a reasonable clustering?

We address both these issues in this section, as they are significant for large high-dimensional data sets. The problem of estimating κ_h is analyzed in Section 4.1. In Section 4.2 we show that hard assignments can reduce computations significantly while giving a reasonable clustering.

4.1 Approximating κ

Recall that because of the lack of an analytical solution, it is not possible to directly estimate the κ values (see (2.5) and (3.6)). One may employ a nonlinear root-finder for estimating κ, but for high dimensional data, problems of overflows and numerical instabilities plague such root-finders. Therefore, an asymptotic approximation of κ is the best choice for estimating κ. Such approaches also have the benefit of taking constant computation time as opposed to any iterative method.

Mardia and Jupp (2000) provided approximations for estimating κ for a single component (2.5), for two limiting cases (Approximations (10.3.7) and (10.3.10) of Mardia and Jupp (2000, pp. 198)):

$$\hat{\kappa} \approx \frac{d-1}{2(1-\bar{r})} \qquad \text{valid for large } \bar{r}, \tag{4.1}$$

$$\hat{\kappa} \approx d\bar{r}\left(1 + \frac{d}{d+2}\bar{r}^2 + \frac{d^2(d+8)}{(d+2)^2(d+4)}\bar{r}^4\right) \qquad \text{valid for small } \bar{r}, \tag{4.2}$$

where $\bar{r}$ is given by (2.5).

These approximations assume that $\kappa \gg d$, which is typically not valid for high dimensions (see the discussion in Section 7 for an intuition). Also, the $\bar{r}$ values corresponding to the text and gene expression data sets considered in this paper are in the mid-range rather than in the two extreme ranges of $\bar{r}$ that are catered to by the above approximations. We obtain a more accurate approximation for κ as described below. With $A_d(\kappa) = \frac{I_{d/2}(\kappa)}{I_{d/2-1}(\kappa)}$, observe that $A_d(\kappa)$ is a ratio of

Bessel functions that differ in their order by just one. Fortunately there exists a continued fraction representation of $A_d(\kappa)$ (Watson, 1995) given by

$$A_d(\kappa) = \frac{I_{d/2}(\kappa)}{I_{d/2-1}(\kappa)} = \cfrac{1}{\frac{d}{\kappa} + \cfrac{1}{\frac{d+2}{\kappa} + \cdots}} . \tag{4.3}$$

Letting $A_d(\kappa) = \bar{r}$ we can write (4.3) approximately as

$$\frac{1}{\bar{r}} \approx \frac{d}{\kappa} + \bar{r} ,$$

which gives the approximation,

$$\kappa \approx \frac{d\bar{r}}{1-\bar{r}^2} .$$

We empirically found (see Section A.3 for details) that the quality of the above approximation can be improved by adding a correction term of $-\bar{r}^3/(1-\bar{r}^2)$ to it. Thus we finally get

$$\hat{\kappa} = \frac{\bar{r}d - \bar{r}^3}{1-\bar{r}^2} . \tag{4.4}$$

The approximation in (4.4) could perhaps be made even more accurate by adding other correction terms that are functions of $\bar{r}$ and d.[3] For other approximations of κ (including the derivations of (4.1) and (4.2)) and some related issues, the reader is referred to the detailed exposition in Dhillon and Sra (2003).

To properly assess the quality of our approximation and compare it with (4.1) and (4.2), first note that a particular value of $\bar{r}$ may correspond to many different combinations of κ and d values. Thus, one needs to evaluate the accuracy of the approximations over the parts of the d-κ plane that are expected to be encountered in the target application domains. Section A.3 of the Appendix provides such an assessment by comparing performances over different slices of the d-κ plane and over a range of $\bar{r}$ values. Below we simply compare the accuracies at a scattering of points on this plane via Table 1 which shows the actual numerical values of κ that the three approximations (4.1), (4.2), and (4.4) yielded at these points. The $\bar{r}$ values shown in the table were computed using (2.5).

$(d, \bar{r}, \kappa)$	$\hat{\kappa}$ = Eq. (4.1)	$\hat{\kappa}$ = Eq. (4.2)	$\hat{\kappa}$ = Eq. (4.4)
$(10, 0.633668, 10)$	12.2839	9.36921	**10.1631**
$(100, 0.46945, 60)$	93.2999	59.3643	**60.0833**
$(500, 0.46859, 300)$	469.506	296.832	**300.084**
$(1000, 0.554386, 800)$	1120.92	776.799	**800.13**

Table 1: Approximations $\hat{\kappa}$ for a sampling of κ and d values.

3. Note that if one wants a more accurate approximation, it is easier to use (4.4) as a starting point and then perform Newton-Raphson iterations for solving $A_d(\hat{\kappa}) - \bar{r} = 0$, since it is easy to evaluate $A_d'(\kappa) = 1 - A_d(\kappa)^2 - \frac{d-1}{\kappa}A_d(\kappa)$. However, for high-dimensional data, accurately computing $A_d(\kappa)$ can be quite slow compared to efficiently approximating $\hat{\kappa}$ using (4.4).

4.2 Analysis of Hard Assignments

In this subsection, we show that hard assignments should give a reasonable clustering in terms of the log-likelihood since they actually maximize a tight lower bound on the incomplete log-likelihood of the data. This result is applicable to any mixture model learning using EM, but the practical advantage in terms of lower computational demands seems to be more substantial when using mixtures of vMFs. The advantages are derived from the various facts outlined below:

- First, note that the partition function, $\sum_{l=1}^{k} \alpha_l f_l(\boldsymbol{x}_i|\theta_l)$, for every data point $\boldsymbol{x}_i$ need not be computed for hard-assignments. This may not be a significant difference for several other models, but this is quite important for vMF distributions. Since the normalization terms $c_d(\kappa_h)$ in $f_h(\boldsymbol{x}_i|\theta_h)$ involve Bessel functions, any reasonable implementation of the algorithm has to employ high-precision representation to avoid under- and over-flows. As a result, computing the partition function is computationally intensive. For hard assignments, this computation is not required resulting in substantially faster running times. In particular, hard-assignments need $O(k)$ computations in high-precision per iteration simply to compute the normalization terms $c_d(\kappa_h), h = 1,\ldots,k$. On the other hand, soft-assignments need $O(nk)$ computations in high-precision per iteration for all $f_l(\boldsymbol{x}_i|\theta_l)$ so that the partition function $\sum_{l=1}^{k} \alpha_l f_l(\boldsymbol{x}_i|\theta_l)$ and the probabilities $p(h|\boldsymbol{x}_i,\Theta)$ can be accurately computed.

- A second issue is regarding the space complexity. Since soft assignments compute all the conditional probabilities, the algorithm has to maintain nk floating point numbers at a desired level of precision. On the other hand, hard assignments only need to maintain the cluster assignments of each point, i.e., n integers. This issue can become critical for large data sets and large number of clusters.

Hence, a hard assignment scheme is often computationally more efficient and scalable both in terms of time and space complexity.

We begin by investigating certain properties of hard-assignments. Hard-assignments have seen extensively used in the statistics (Coleman et al., 1999; McLachlan and Peel, 2000) as well as machine learning literature (Kearns et al., 1997; Banerjee et al., 2004). In statistics, the hard assignment approach is better known as classification maximum likelihood approach (McLachlan, 1982). Although soft-assignments are theoretically well motivated (Collins, 1997; Neal and Hinton, 1998), hard-assignments have not received much theoretical attention with some notable exceptions (Kearns et al., 1997). However, algorithms employing hard-assignments, being computationally more efficient especially for large data sets, are often typically more practical than algorithms that use soft-assignments. Hence it is worthwhile to examine the behavior of hard-assignments from a theoretical perspective. In the rest of this section, we formally study the connection between soft and hard-assignments in the EM framework.

The distribution q in (3.8) belongs to the class $\mathcal{H}$ of probability distributions that assume probability value 1 for some mixture component and 0 for all others. In the hard assignment setting, the hidden random variables are restricted to have distributions that are members of $\mathcal{H}$. Since $\mathcal{H}$ is a subset of all possible distributions on the events, for a typical mixture model the distribution following (3.7) will not belong to this subset. The important question is: Is there a way to optimally pick a distribution from $\mathcal{H}$, perform a regular M-step, and guarantee that the incomplete log-likelihood of the data does not decrease? Unfortunately, such a way may not exist in general. However, it is possible to reasonably lower bound the incomplete log-likelihood of the data using expectations over an *optimal* distribution $q \in \mathcal{H}$, as elucidated below. Thus, clustering using hard-assignments essentially maximizes a lower bound on the incomplete log-likelihood.

We now show that the expectation over q is a reasonable lower bound on the incomplete log-likelihood of the data in the sense that the expectation over q is itself lower bounded by the expectation of the complete log-likelihood (3.2) over the distribution p given by (3.7). Further, we show that q as given by (3.8) gives the tightest lower bound among all distributions in $\mathcal{H}$.

Following the arguments of Neal and Hinton (1998), we introduce the function $F(\tilde{p}, \Theta)$ given by

$$F(\tilde{p}, \Theta) = E_{\tilde{p}}[\ln P(\mathcal{X}, \mathcal{Z}|\Theta)] + H(\tilde{p}), \tag{4.5}$$

where $H(\tilde{p})$ gives the Shannon entropy of a discrete distribution $\tilde{p}$. The E- and the M-steps of the EM algorithm can be shown to *alternately maximize* the function F. In the E-step, for a given value of Θ, the distribution $\tilde{p}$ is chosen to maximize $F(\tilde{p}, \Theta)$ for that Θ, and, in the M-step, for a given value of $\tilde{p}$, the parameters Θ are estimated to maximize $F(\tilde{p}, \Theta)$ for the given $\tilde{p}$. Consider p given by (3.7). It can be shown (Neal and Hinton, 1998) that for a given Θ, this value of p is optimal, i.e, $p = \text{argmax}_{\tilde{p}} F(\tilde{p}, \Theta)$. Then,

$$\begin{aligned} F(p, \Theta) &= E_p[\ln P(\mathcal{X}, \mathcal{Z}|\Theta)] + H(p) \\ &= E_p[\ln P(\mathcal{X}, \mathcal{Z}|\Theta)] - E_p[\ln P(\mathcal{Z}|(\mathcal{X}, \Theta))] \\ &= E_p\left[\ln\left(\frac{P(\mathcal{X}, \mathcal{Z}|\Theta)}{P(\mathcal{Z}|(\mathcal{X}, \Theta))}\right)\right] = E_p[\ln P(\mathcal{X}|\Theta)] \\ &= \ln P(\mathcal{X}|\Theta). \end{aligned} \tag{4.6}$$

Since (3.7) gives the optimal choice of the distribution, the functional value of F is smaller for any other choice of $\tilde{p}$. In particular, if $\tilde{p} = q$ as in (3.8), we have

$$F(q, \Theta) \leq F(p, \Theta) = \ln P(\mathcal{X}|\Theta).$$

Since $H(q) = 0$, from (4.5) we have

$$E_q[\ln P(\mathcal{X}, \mathcal{Z}|\Theta)] \leq \ln P(\mathcal{X}|\Theta). \tag{4.7}$$

Thus, the expectation over q actually lower bounds the likelihood of the data. We go one step further to show that this is in fact a reasonably tight lower bound in the sense that the expectation over q is lower bounded by the expectation over p of the complete data log-likelihood. To this end, we first prove the following result.

Lemma 1 *If p is given by (3.7) and q is given by (3.8) then,*

$$E_p[\ln P(\mathcal{Z}|(\mathcal{X}, \Theta))] \leq E_q[\ln P(\mathcal{Z}|(\mathcal{X}, \Theta))].$$

Proof Let $h_i^* = \underset{h}{\operatorname{argmax}}\ p(h|\boldsymbol{x}_i,\Theta)$. Then, $p(h|\boldsymbol{x}_i,\Theta) \leq p(h_i^*|\boldsymbol{x}_i,\Theta), \forall h$. Using the definitions of p and q, we have

$$\begin{aligned}
E_p[\ln P(\mathcal{Z}|(\mathcal{X},\Theta))] &= \sum_{i=1}^{n}\sum_{h=1}^{k} p(h|\boldsymbol{x}_i,\Theta)\ln p(h|\boldsymbol{x}_i,\Theta)\\
&\leq \sum_{i=1}^{n}\sum_{h=1}^{k} p(h|\boldsymbol{x}_i,\Theta)\ln p(h_i^*|\boldsymbol{x}_i,\Theta)\\
&= \sum_{i=1}^{n}\ln p(h_i^*|\boldsymbol{x}_i,\Theta)\sum_{h=1}^{k} p(h|\boldsymbol{x}_i,\Theta) \;=\; \sum_{i=1}^{n}\ln p(h_i^*|\boldsymbol{x}_i,\Theta)\\
&= \sum_{i=1}^{n}\sum_{h=1}^{k} q(h|\boldsymbol{x}_i,\Theta)\ln p(h|\boldsymbol{x}_i,\Theta)\\
&= E_q[\ln P(\mathcal{Z}|(\mathcal{X},\Theta))]. \qquad\blacksquare
\end{aligned}$$

Now, adding the incomplete data log-likelihood to both sides of the inequality proven above, we obtain

$$\begin{aligned}
E_p[\ln P(\mathcal{Z}|(\mathcal{X},\Theta))] + \ln P(\mathcal{X}|\Theta) &\leq E_q[\ln P(\mathcal{Z}|(\mathcal{X},\Theta))] + \ln P(\mathcal{X}|\Theta),\\
E_p[\ln(P(\mathcal{Z}|(\mathcal{X},\Theta))P(\mathcal{X}|\Theta))] &\leq E_q[\ln(P(\mathcal{Z}|(\mathcal{X},\Theta))P(\mathcal{X}|\Theta))],\\
E_p[\ln P(\mathcal{X},\mathcal{Z}|\Theta)] &\leq E_q[\ln P(\mathcal{X},\mathcal{Z}|\Theta)]. \qquad (4.8)
\end{aligned}$$

From, (4.7) and (4.8), we infer

$$E_p[\ln P(\mathcal{X},\mathcal{Z}|\Theta)] \leq E_q[\ln P(\mathcal{X},\mathcal{Z}|\Theta)] \leq \ln P(\mathcal{X}|\Theta).$$

Let $\tilde{q}$ be any other distribution in the class of distributions $\mathcal{H}$ with $\tilde{q}(\tilde{h}_i|\boldsymbol{x}_i,\Theta) = 1$ and $\tilde{q}(h|\boldsymbol{x}_i,\Theta = 0)$ for $h \neq \tilde{h}_i$. Then,

$$\begin{aligned}
E_{\tilde{q}}[\ln P(\mathcal{Z}|\mathcal{X},\Theta)] &= \sum_{i=1}^{n}\sum_{h=1}^{k} \tilde{q}(h|\boldsymbol{x}_i,\Theta)\ln p(h|\boldsymbol{x}_i,\Theta) \;=\; \sum_{i=1}^{n}\ln p(\tilde{h}_i|\boldsymbol{x}_i,\Theta)\\
&\leq \sum_{i=1}^{n}\ln p(h_i^*|\boldsymbol{x}_i,\Theta) \;=\; \sum_{i=1}^{n}\sum_{h=1}^{k} q(h|\boldsymbol{x}_i,\Theta)\ln p(h|\boldsymbol{x}_i,\Theta)\\
&= E_q[\ln P(\mathcal{Z}|\mathcal{X},\Theta)].
\end{aligned}$$

Hence, the choice of q as in (3.8) is optimal. This analysis forms the basis of the `hard-moVMF` algorithm presented in the next section.

5. Algorithms

The developments of the previous section naturally lead to two algorithms for clustering directional data. The algorithms are centered on soft and hard-assignment schemes and are titled `soft-moVMF` and `hard-moVMF` respectively. The `soft-moVMF` algorithm (Algorithm 1) estimates the parameters of the mixture model exactly following the derivations in Section 3 using EM. Hence, it assigns soft (or probabilistic) labels to each point that are given by the posterior probabilities of the components

```
Algorithm 1 soft-moVMF
Input: Set $\mathcal{X}$ of data points on $\mathbb{S}^{d-1}$
Output: A soft clustering of $\mathcal{X}$ over a mixture of $k$ vMF distributions
  Initialize all $\alpha_h, \boldsymbol{\mu}_h, \kappa_h,\ h = 1, \cdots, k$
  repeat
    {The E (Expectation) step of EM}
    for $i = 1$ to $n$ do
      for $h = 1$ to $k$ do
        $f_h(\boldsymbol{x}_i|\theta_h) \leftarrow c_d(\kappa_h) e^{\kappa_h \boldsymbol{\mu}_h^T \boldsymbol{x}_i}$
      end for
      for $h = 1$ to $k$ do
        $p(h|\boldsymbol{x}_i, \Theta) \leftarrow \frac{\alpha_h f_h(\boldsymbol{x}_i|\theta_h)}{\sum_{l=1}^k \alpha_l f_l(\boldsymbol{x}_i|\theta_l)}$
      end for
    end for
    {The M (Maximization) step of EM}
    for $h = 1$ to $k$ do
      $\alpha_h \leftarrow \frac{1}{n} \sum_{i=1}^n p(h|\boldsymbol{x}_i, \Theta)$
      $\boldsymbol{\mu}_h \leftarrow \sum_{i=1}^n \boldsymbol{x}_i p(h|\boldsymbol{x}_i, \Theta)$
      $\bar{r} \leftarrow \|\boldsymbol{\mu}_h\| / (n\alpha_h)$
      $\boldsymbol{\mu}_h \leftarrow \boldsymbol{\mu}_h / \|\boldsymbol{\mu}_h\|$
      $\kappa_h \leftarrow \frac{\bar{r}d - \bar{r}^3}{1 - \bar{r}^2}$
    end for
  until convergence
```

of the mixture conditioned on the point. On termination, the algorithm gives the parameters $\Theta = \{\alpha_h, \boldsymbol{\mu}_h, \kappa_h\}_{h=1}^k$ of the k vMF distributions that model the data set $\mathcal{X}$, as well as the *soft-clustering*, i.e., the posterior probabilities $p(h|\boldsymbol{x}_i, \Theta)$, for all h and i.

The hard-moVMF algorithm (Algorithm 2) estimates the parameters of the mixture model using a hard assignment, or, *winner takes all* strategy. In other words, we do the assignment of the points based on a derived posterior distribution given by (3.8). After the hard assignments in every iteration, each point *belongs* to a single cluster. As before, the updates of the component parameters are done using the posteriors of the components, given the points. The crucial difference in this case is that the posterior probabilities are allowed to take only binary (0/1) values. Upon termination, Algorithm 2 yields a hard clustering of the data and the parameters $\Theta = \{\alpha_h, \boldsymbol{\mu}_h, \kappa_h\}_{h=1}^k$ of the k vMFs that model the input data set $\mathcal{X}$.

5.1 Revisiting Spherical Kmeans

In this section we show that upon enforcing certain restrictive assumptions on the generative model, the spkmeans algorithm (Algorithm 3) can be viewed as a special case of both the soft-moVMF and hard-moVMF algorithms.

More precisely, assume that in our mixture of vMFs, the priors of all the components are equal, i.e., $\alpha_h = 1/k$ for all h. Further assume that all the components have (equal) infinite concentration parameters, i.e., $\kappa_h = \kappa \to \infty$ for all h. Under these assumptions the E-step in the soft-moVMF

Algorithm 2 `hard-moVMF`

```
Input: Set $\mathcal{X}$ of data points on $\mathbb{S}^{d-1}$
Output: A disjoint $k$-partitioning of $\mathcal{X}$
  Initialize all $\alpha_h, \boldsymbol{\mu}_h, \kappa_h,\ h = 1, \cdots, k$
  repeat
    {The Hardened E (Expectation) step of EM}
    for $i = 1$ to $n$ do
      for $h = 1$ to $k$ do
        $f_h(\boldsymbol{x}_i|\theta_h) \leftarrow c_d(\kappa_h) e^{\kappa_h \boldsymbol{\mu}_h^T \boldsymbol{x}_i}$
      end for
      for $h = 1$ to $k$ do
        $q(h|\boldsymbol{x}_i, \Theta) \leftarrow \begin{cases} 1, & \text{if } h = \underset{h'}{\text{argmax}}\ \alpha_{h'}\ f_{h'}(\boldsymbol{x}_i|\theta_{h'}) \\ 0, & \text{otherwise.} \end{cases}$
      end for
    end for
    {The M (Maximization) step of EM}
    for $h = 1$ to $k$ do
      $\alpha_h \leftarrow \frac{1}{n}\sum_{i=1}^{n} q(h|\boldsymbol{x}_i, \Theta)$
      $\boldsymbol{\mu}_h \leftarrow \sum_{i=1}^{n} \boldsymbol{x}_i q(h|\boldsymbol{x}_i, \Theta)$
      $\bar{r} \leftarrow \|\boldsymbol{\mu}_h\|/(n\alpha_h)$
      $\boldsymbol{\mu}_h \leftarrow \boldsymbol{\mu}_h/\|\boldsymbol{\mu}_h\|$
      $\kappa_h \leftarrow \frac{\bar{r}d - \bar{r}^3}{1 - \bar{r}^2}$
    end for
  until convergence.
```

algorithm reduces to assigning a point to its *nearest* cluster, where nearness is computed as a cosine similarity between the point and the cluster representative. Thus, a point $\boldsymbol{x}_i$ will be assigned to cluster $h^* = \underset{h}{\text{argmax}}\ \boldsymbol{x}_i^T \boldsymbol{\mu}_h$, since

$$p(h^*|\boldsymbol{x}_i, \Theta) = \lim_{\kappa \to \infty} \frac{e^{\kappa\ \boldsymbol{x}_i^T \boldsymbol{\mu}_{h^*}}}{\sum_{h=1}^{k} e^{\kappa\ \boldsymbol{x}_i^T \boldsymbol{\mu}_h}} = 1,$$

and $p(h|\boldsymbol{x}_i, \Theta) \to 0$, as $\kappa \to \infty$ for all $h \neq h^*$.

To show that `spkmeans` can also be seen as a special case of the `hard-moVMF`, in addition to assuming the priors of the components to be equal, we further assume that the concentration parameters of all the components are equal, i.e., $\kappa_h = \kappa$ for all h. With these assumptions on the model, the estimation of the common concentration parameter becomes unessential since the hard assignment will depend only on the value of the cosine similarity $\boldsymbol{x}_i^T \boldsymbol{\mu}_h$, and `hard-moVMF` reduces to `spkmeans`.

In addition to the abovementioned algorithms, we report experimental results on another algorithm `fskmeans` (Banerjee and Ghosh, 2002) that belongs to the same class in the sense that, like `spkmeans`, it can be derived from the mixture of vMF models with some restrictive assumptions. In `fskmeans`, the centroids of the mixture components are estimated as in `hard-movMF`. The κ value

Algorithm 3 `spkmeans`

```
Input: Set $\mathcal{X}$ of data points on $\mathbb{S}^{d-1}$
Output: A disjoint $k$-partitioning $\{\mathcal{X}_h\}_{h=1}^k$ of $\mathcal{X}$
  Initialize $\boldsymbol{\mu}_h,\ h = 1, \cdots, k$
  repeat
    {The E (Expectation) step of EM}
    Set $\mathcal{X}_h \leftarrow \emptyset,\ h = 1, \cdots, k$
    for $i = 1$ to $n$ do
      $\mathcal{X}_h \leftarrow \mathcal{X}_h \cup \{\boldsymbol{x}_i\}$ where $h = \underset{h'}{\operatorname{argmax}}\ \boldsymbol{x}_i^T \boldsymbol{\mu}_{h'}$
    end for
    {The M (Maximization) step of EM}
    for $h = 1$ to $k$ do
      $\boldsymbol{\mu}_h \leftarrow \dfrac{\sum_{\boldsymbol{x} \in \mathcal{X}_h} \boldsymbol{x}}{\|\sum_{\boldsymbol{x} \in \mathcal{X}_h} \boldsymbol{x}\|}$
    end for
  until convergence.
```

for a component is *explicitly set* to be inversely proportional to the number of points in the cluster corresponding to that component. This explicit choice simulates a frequency sensitive competitive learning that implicitly prevents the formation of null clusters, a well-known problem in regular kmeans (Bradley et al., 2000).

6. Experimental Results

We now offer some experimental validation to assess the quality of clustering results achieved by our algorithms. We compare the following four algorithms on numerous data sets.

1. Spherical KMeans (Dhillon and Modha, 2001)—`spkmeans`.
2. Frequency Sensitive Spherical KMeans (Banerjee and Ghosh, 2002)—`fskmeans`.
3. moVMF based clustering using hard assignments (Section 3)—`hard-moVMF`.
4. moVMF based clustering using soft assignments (Section 3)—`soft-moVMF`.

It has already been established that `kmeans` using Euclidean distance performs much worse than `spkmeans` for text data (Strehl et al., 2000), so we do not consider it here. Generative model based algorithms that use mixtures of Bernoulli or multinomial distributions, which have been shown to perform well for text data sets, have also not been included in the experiments. This exclusion is done as a recent empirical study over 15 text data sets showed that simple versions of vMF mixture models (with κ constant for all clusters) outperform the multinomial model except for only one data set (Classic3), and the Bernoulli model was inferior for all data sets (Zhong and Ghosh, 2003b).

6.1 Data Sets

The data sets that we used for empirical validation and comparison of our algorithms were carefully selected to represent some typical clustering problems. We also created various subsets of some of the data sets for gaining greater insight into the nature of clusters discovered or to model some particular clustering scenario (e.g., balanced clusters, skewed clusters, overlapping clusters etc.). We drew our data from five sources: Simulation, Classic3, Yahoo News, CMU 20 Newsgroup and Yeast Gene Expressions. For all the text document data sets, the toolkit MC (Dhillon et al., 2001) was used for creating a high-dimensional vector space model that each of the four algorithms utilized. MATLAB code was used to render the input as a vector space for both the simulated and gene-expression data sets.

- **Simulation.** We use simulated data to verify that the discrepancy between computed values of the parameters and their true values is small. Our simulated data serves the principal purpose of validating the "correctness" of our implementations. We used a slight modification of the algorithm given by Wood (1994) to generate a set of data points following a given vMF distribution. We describe herein, two synthetic data sets. The first data set **small-mix** is 2-dimensional and is used to illustrate soft-clustering. The second data set **big-mix** is a high-dimensional data set that could serve as a model for real world text data sets. Let the triple (n,d,k) denote the number of sample points, the dimensionality of a sample point and the number of clusters respectively.

    1. **small-mix:** This data has $(n,d,k) = (50,2,2)$. The mean direction of each component is a random unit vector. Each component has $\kappa = 4$.
    2. **big-mix:** This data has $(n,d,k) = (5000,1000,4)$. The mean direction of each component is a random unit vector, and the κ values of the components are 650.98, 266.83, 267.83, and 612.88. The mixing weights for each component are 0.251, 0.238, 0.252, and 0.259.

- **Classic3.** Classic3 is a well known collection of documents. It is an easy data set to cluster since it contains documents from three well-separated sources. Moreover, the intrinsic clusters are largely balanced.

    1. **Classic3:** This corpus contains 3893 documents, among which 1400 CRANFIELD documents are from aeronautical system papers, 1033 MEDLINE documents are from medical journals, and 1460 CISI documents are from information retrieval papers. The particular vector space model used had a total of 4666 features (words). Thus each document, after normalization, is represented as a unit vector in a 4666-dimensional space.
    2. **Classic300:** Classic300 is a subset of the Classic3 collection and has 300 documents. From each category of Classic3, we picked 100 documents at random to form this particular data set. The dimensionality of the data was 5471.[4]
    3. **Classic400:** Classic400 is a subset of Classic3 that has 400 documents. This data set has 100 randomly chosen documents from the MEDLINE and CISI categories and 200 randomly chosen documents from the CRANFIELD category. This data set is specifically designed to create unbalanced clusters in an otherwise easily separable and balanced data set. The dimensionality of the data was 6205.

4. Note that the dimensionality in Classic300 is larger than the that of Classic3. Although the same options were used in the MC toolkit for word pruning, due to very different words distributions, fewer words got prunned for Classic300 in the 'too common' or 'too rare' categories.

- **Yahoo News (K-series).** This compilation has 2340 Yahoo news articles from 20 different categories. The underlying clusters in this data set are highly skewed in terms of the number of documents per cluster, with sizes ranging from 9 to 494. The skewness presents additional challenges to clustering algorithms.

- **CMU Newsgroup.** The CMU Newsgroup data set is a well known compilation of documents (Newsgroups). We tested our algorithms on not only the original data set, but on a variety of subsets with differing characteristics to explore and understand the behavior of our algorithms.

  1. **News20:** This standard data set is a collection of 19,997 messages, gathered from 20 different USENET newsgroups. One thousand messages are drawn from the first 19 newsgroups, and 997 from the twentieth. The headers for each of the messages are then removed to avoid biasing the results. The particular vector space model used had 25924 words. News20 embodies the features characteristic of a typical text data set—high-dimensionality, sparsity and significantly overlapping clusters.
  2. **Small-news20:** We formed this set by selecting 2000 messages from original News20 data set. We randomly selected 100 messages from each category in the original data set. Hence this data set has balanced classes (though there may be overlap). The dimensionality of the data was 13406.
  3. **Same-100/1000** is a collection of 100/1000 messages from 3 very similar newsgroups: comp.graphics, comp.os.ms-windows, comp.windows.x.
  4. **Similar-100/1000** is a collection of 100/1000 messages from 3 somewhat similar newsgroups: talk.politics.guns, talk.politics.mideast, talk.politics.misc.
  5. **Different-100/1000** is a collection of 100/1000 messages from 3 very different newsgroups: alt.atheism, rec.sport.baseball, sci.space.

- **Yeast Gene Expressions.** Gene-expression data was selected to offer a clustering domain different from text analysis. As previously motivated, the use of Pearson correlation for the analysis of gene expression data is common, so a directional model is well-suited. Coincident to this domain are the difficulties of cluster validation because of the unavailability of true labels. Such difficulties make the gene expression data a more challenging and perhaps a more rewarding domain for clustering.

  Gene expression data is presented as a matrix of genes (rows) by expression values (columns). The expression vectors are constructed using DNA microarray experiments. We used a subset of the Rosetta Inpharmatics yeast gene expression set (Hughes et al., 2000). The original data set consists of 300 experiments measuring expression of 6,048 yeast genes. Out of these we selected a subset of 996 genes for clustering (Dhillon et al., 2002b). For each of the 996 genes the 300-element expression vector was normalized to have unit Euclidean (L_2) norm.

6.2 Methodology

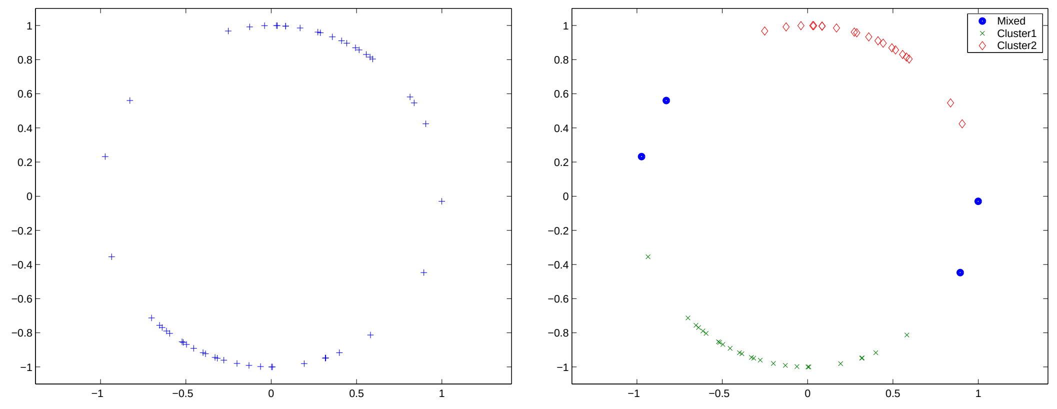
Except for the gene expression data set, performance of the algorithms on all the data sets has been analyzed using *mutual information* (MI) between the cluster and class labels. For gene data, due to the absence of true labels, we have to take recourse to reporting some internal figures of merit. We defer a discussion of the same to Section 6.7.

The MI gives the amount of statistical similarity between the cluster and class labels (Cover and Thomas, 1991). If X is a random variable for the cluster assignments and Y is a random variable for the pre-existing labels on the same data, then their MI is given by $I(X;Y) = E[\ln \frac{p(X,Y)}{p(X)p(Y)}]$ where the expectation is computed over the joint distribution of (X,Y) estimated from a particular clustering of the data set under consideration. For `soft-moVMF` we "harden" the clustering produced by labeling a point with the cluster label for which it has the highest value of posterior probability (ties broken arbitrarily), in order to evaluate MI. Note that variants of MI have been used to evaluate clustering algorithms by several researchers. Meilă (2003) used a related concept called variation of information to compare clusterings. An MDL-based formulation that uses the MI between cluster assignments and class labels was proposed by Dom (2001).

All results reported herein have been averaged over 10 runs. All algorithms were started with the same random initialization to ensure fairness of comparison. Each run was started with a *different* random initialization. However, no algorithm was restarted within a given run and all of them were allowed to run to completion. Since the standard deviations of MI were reasonably small for all algorithms, to reduce clutter, we have chosen to omit a display of error bars in our plots. Also, for practical reasons, the estimate of κ was upper bounded by a large number (10^4, in this case) in order to prevent numeric overflows. For example, during the iterations, if a cluster has only one point, the estimate of κ will be infinity (a divide by zero error). Upper bounding the estimate is similar in flavor to ensuring the estimated covariance of a multi-variate Gaussian in a mixture of Gaussians to be non-singular.

6.3 Simulated Data Sets

First, to build some intuition and confidence in the working of our vMF based algorithms we exhibit relevant details of `soft-moVMF`'s behavior on the small-mix data set shown in Figure 1(a).



(a)
The small-mix data set.

(b)
A clustering of small-mix.

Figure 1: Small-mix data set and its clustering by `soft-moVMF`.

The clustering produced by our soft cluster assignment algorithm is shown in Figure 1(b). The four points (taken clockwise) marked with solid circles have cluster labels $(0.15, 0.85)$, $(0.77, 0.23)$, $(.82, .18)$ and $(.11, .89)$, where a cluster label $(p, 1-p)$ for a point means that the point has probability p of belonging to Cluster 1 and probability $1-p$ of belonging to Cluster 2. All other points are categorized to belong to a single cluster by ignoring small (less than 0.10) probability values.

The confusion matrix, obtained by "hardening" the clustering produced by `soft-moVMF` for the small-mix data set is $\begin{bmatrix} 26 & 1 \\ 0 & 23 \end{bmatrix}$. As is evident from this confusion matrix, the clustering performed by `soft-moVMF` is excellent, though not surprising, since small-mix is a data set with well-separated clusters. Further testimony to `soft-moVMF`'s performance is served by Table 2, which shows the discrepancy between true and estimated parameters for the small-mix collection.

Cluster	$\boldsymbol{\mu}$	$\hat{\boldsymbol{\mu}}$	κ	$\hat{\kappa}$	α	$\hat{\alpha}$
1	(-0.251, -0.968)	(-0.279, -0.960)	4	3.78	0.48	0.46
2	(0.399, 0.917)	(0.370, 0.929)	4	3.53	0.52	0.54

Table 2: True and estimated parameters for small-mix using `soft-moVMF`.

In the table $\boldsymbol{\mu}, \kappa, \alpha$ represent the true parameters and $\hat{\boldsymbol{\mu}}, \hat{\kappa}, \hat{\alpha}$ represent the estimated parameters. We can see that even in the presence of a limited number of data points in the small-mix data set (50 points), the estimated parameters approximate the true parameters quite well.

Before moving onto real data sets let us briefly look at the behavior of the algorithms on the larger data set big-mix. On calculating MI as described previously we found that all the algorithms performed similarly with MI values close to one. We attribute this good performance of all the algorithms to the availability of a sufficient number of data points and similar sized clusters. For reference Table 3 offers numerical evidence about the performance of `soft-moVMF` on the big-mix data set.

$\min \boldsymbol{\mu}^T \hat{\boldsymbol{\mu}}$	$\text{avg}\, \boldsymbol{\mu}^T \hat{\boldsymbol{\mu}}$	$\max \frac{\lvert\kappa-\hat{\kappa}\rvert}{\lvert\kappa\rvert}$	$\text{avg} \frac{\lvert\kappa-\hat{\kappa}\rvert}{\lvert\kappa\rvert}$	$\max \frac{\lvert\alpha-\hat{\alpha}\rvert}{\lvert\alpha\rvert}$	$\text{avg} \frac{\lvert\alpha-\hat{\alpha}\rvert}{\lvert\alpha\rvert}$
0.994	0.998	0.006	0.004	0.002	0.001

Table 3: Performance of `soft-moVMF` on big-mix data set.

6.4 Classic3 Family of Data Sets

Table 4 shows typical confusion matrices obtained for the full Classic3 data set. We observe that the performance of all the algorithms is quite similar and there is no added advantage yielded by using the general moVMF model as compared to the other algorithms. This observation can be explained by noting that the clusters of Classic3 are well separated and have a sufficient number of documents. For this clustering `hard-moVMF` yielded κ values of $(732.13, 809.53, 1000.04)$, while `soft-moVMF` reported κ values of $(731.55, 808.21, 1002.95)$.

Table 5 shows the confusion matrices obtained for the Classic300 data set. Even though Classic300 is well separated, the small number of documents per cluster makes the problem somewhat difficult for `fskmeans` and `spkmeans`, while `hard-moVMF` has a much better performance due to its model flexibility. The `soft-moVMF` algorithm performs appreciably better than the other three algorithms.

It seems that the low number of documents does not pose a problem for `soft-moVMF` and it ends up getting an almost perfect clustering for this data set. Thus in this case, despite the low number

`fskmeans`			`spkmeans`			`hard-moVMF`			`soft-moVMF`		
med	cisi	cran	med	cisi	cran	med	cisi	cran	med	cisi	cran
1019	0	0	**1019**	0	0	**1018**	0	0	**1019**	0	1
1	6	**1386**	1	6	**1386**	2	6	**1387**	1	4	**1384**
13	**1454**	12	13	**1454**	12	13	**1454**	11	13	**1456**	13

Table 4: Comparative confusion matrices for 3 clusters of Classic3 (rows represent clusters).

`fskmeans`			`spkmeans`			`hard-moVMF`			`soft-moVMF`		
med	cisi	cran	med	cisi	cran	med	cisi	cran	med	cisi	cran
29	**38**	22	29	**38**	22	3	**72**	1	0	**98**	0
31	27	**38**	31	27	**38**	**62**	28	17	**99**	2	0
40	35	**40**	**40**	35	**40**	35	0	**82**	1	0	**100**

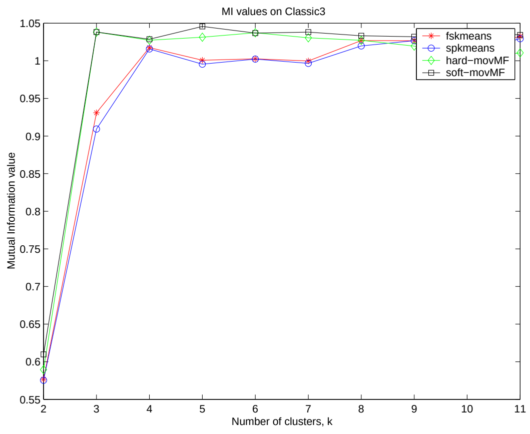
Table 5: Comparative confusion matrices for 3 clusters of Classic300.

of points per cluster, the superior modeling power of our moVMF based algorithms prevents them from getting trapped in inferior local-minima as compared to the other algorithms—resulting in a better clustering.

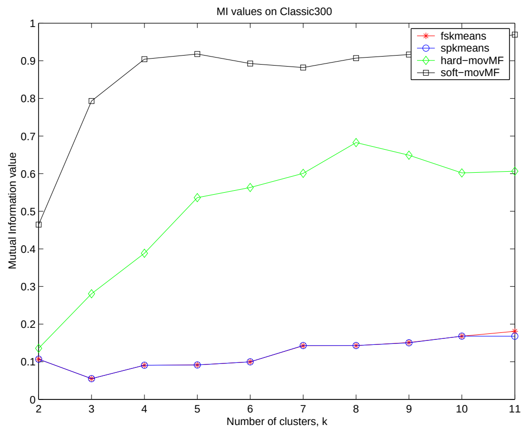
The confusion matrices obtained for the Classic400 data set are displayed in Table 6. The behavior of the algorithms for this data set is quite interesting. As before, due to the small number of documents per cluster, `fskmeans` and `spkmeans` give a rather mixed confusion matrix. The `hard-moVMF` algorithm gets a significant part of the bigger cluster correctly and achieves some amount of separation between the two smaller clusters. The `soft-moVMF` algorithm exhibits a somewhat intriguing behavior. It splits the bigger cluster into two, relatively pure segments, and merges the smaller two into one cluster. When 4 clusters are requested from `soft-moVMF`, it returns 4 very pure clusters (not shown in the confusion matrices) two of which are almost equal sized segments of the bigger cluster.

An engaging insight into the working of the algorithms is provided by considering their clustering performance when they are requested to produce greater than the “natural” number of clusters. In Table 7 we show the confusion matrices resulting from 5 clusters of the Classic3 corpus. The matrices suggest that the moVMF algorithms have a tendency of trying to maintain larger clusters intact as long as possible, and breaking them into reasonably pure and comparably sized parts when they absolutely must. This behavior of our moVMF algorithms coupled with the observations in Table 6, suggest a clustering method in which one could generate a slightly higher number of clusters than required, and then agglomerate them appropriately.

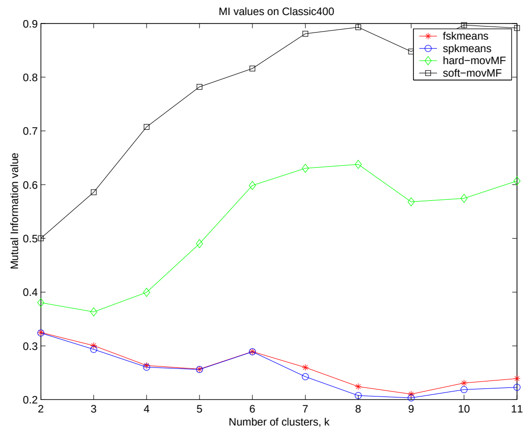
The MI plots for the various Classic3 data sets are given in Figures 2(a)-(c). For the full Classic3 data set (Figure 2(a)), all the algorithms perform almost similarly at the true number of clusters. However, as the number of clusters increases, `soft-moVMF` seems to outperform the others by a significant margin. For Classic300 (Figure 2(b)) and Classic400 (Figure 2(c)), `soft-moVMF` seems to significantly outperform the other algorithms. In fact, for these two data sets, `soft-moVMF` performs substantially better than the other three, even at the correct number of clusters. Among the other three, `hard-moVMF` seems to perform better than `spkmeans` and `fskmeans` across the range of clusters.



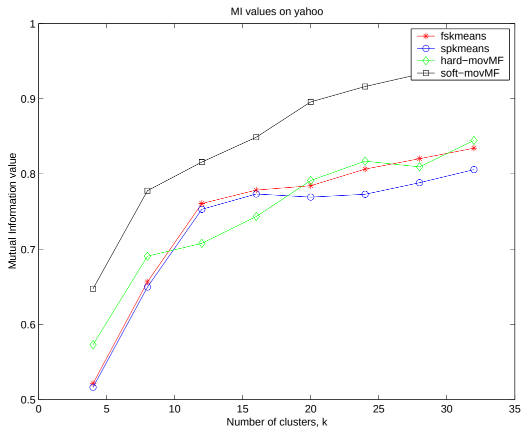
(a) Comparison of MI values for Classic3.



(b) Comparison of MI values for Classic300.



(c) Comparison of MI values for Classic400.



(d) Comparison of MI values for Yahoo20.

Figure 2: Comparison of the algorithms for the Classic3 data sets and the Yahoo News data set.

`fskmeans`			`spkmeans`			`hard-moVMF`			`soft-moVMF`		
med	cisi	cran	med	cisi	cran	med	cisi	cran	med	cisi	cran
27	16	**55**	27	17	**54**	**56**	28	20	0	0	**91**
51	**83**	12	**51**	**82**	12	44	**72**	14	**82**	**99**	2
23	1	**132**	23	1	**133**	1	0	**165**	19	1	**106**

Table 6: Comparative confusion matrices for 3 clusters of Classic400.

`fskmeans`			`spkmeans`			`hard-moVMF`			`soft-moVMF`		
med	cisi	cran	med	cisi	cran	med	cisi	cran	med	cisi	cran
2	4	**312**	2	4	**323**	3	5	**292**	0	1	**1107**
8	**520**	10	8	**512**	9	**511**	1	0	5	**1455**	14
5	**936**	6	5	**944**	6	**514**	1	0	**526**	2	1
1018	0	1	**1018**	0	1	0	2	**1093**	**501**	0	0
0	0	**1069**	0	0	**1059**	5	**1451**	13	1	2	**276**

Table 7: Comparative confusion matrices for 5 clusters of Classic3.

6.5 Yahoo News Data Set

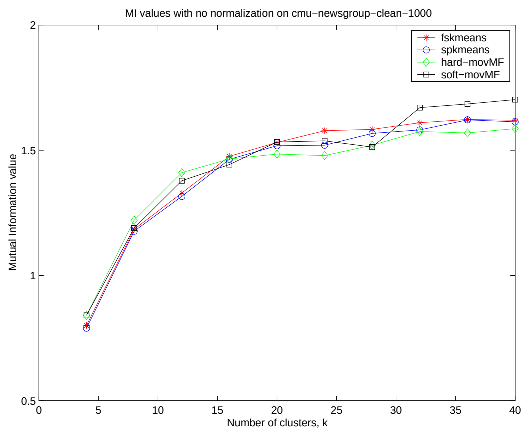
The Yahoo News data set is a relatively difficult data set for clustering since it has a fair amount of overlap among its clusters and the number of points per cluster is low. In addition, the clusters are highly skewed in terms of their comparative sizes.

Results for the different algorithms can be seen in Figure 2(d). Over the entire range, `soft-moVMF` consistently performs better than the other algorithms. Even at the correct number of clusters $k = 20$, it performs significantly better than the other algorithms.

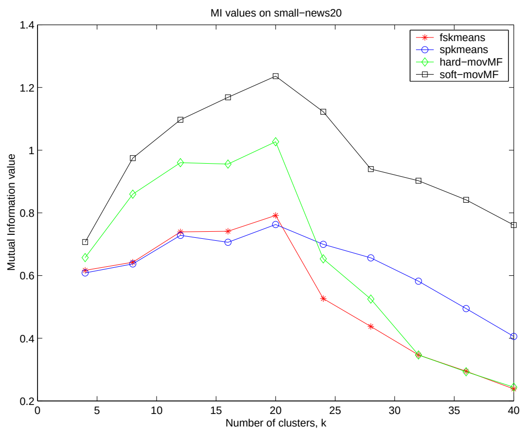
6.6 CMU Newsgroup Family of Data Sets

Now we discuss clustering performance of the four algorithms on the CMU Newsgroup data sets. Figure 3(a) shows the MI plots for the full News20 data set. All the algorithms perform similarly until the true number of clusters after which `soft-moVMF` and `spkmeans` perform better than the others. We do not notice any interesting differences between the four algorithms from this Figure.

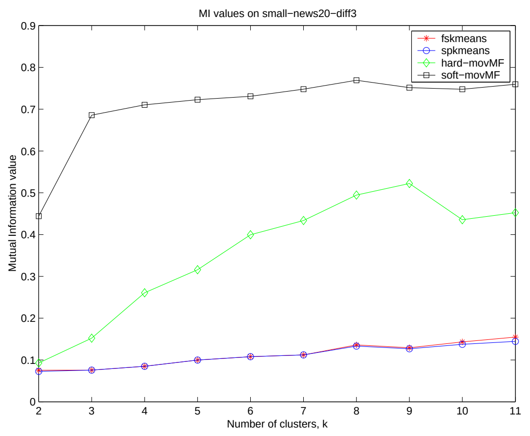
Figure 3(b) shows MI plots for the Small-News20 data set and the results are of course different. Since the number of documents per cluster is small (100), as before `spkmeans` and `fskmeans` do not perform that well, even at the true number of clusters, whereas `soft-moVMF` performs considerably better than the others over the entire range. Again, `hard-moVMF` exhibits good MI values until the true number of clusters, after which it falls sharply. On the other hand, for the data sets that have a reasonably large number of documents per cluster, another kind of behavior is usually observed. All the algorithms perform quite similarly until the true number of clusters, after which `soft-moVMF` performs significantly better than the other three. This behavior can be observed in Figures 3(d), 3(f) and 4(b). We note that the other three algorithms perform quite similarly over the entire range of clusters. We also observe that for an easy data set like Different-1000, the MI values peak at the true number of clusters, whereas for a more difficult data set such as Similar-1000 the MI values increase as the clusters get further refined. This behavior is expected since the clusters in Similar-1000 have much greater overlap than those in Different-1000.



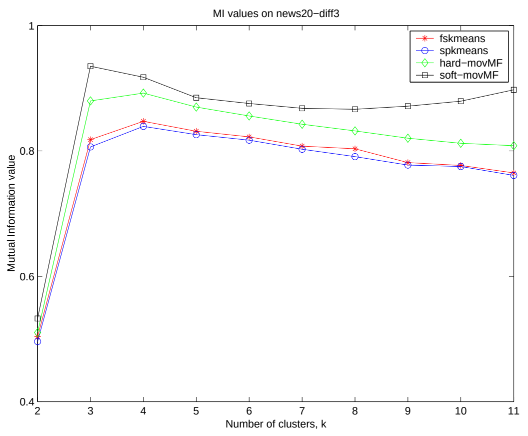
(a) Comparison of MI values for News20.



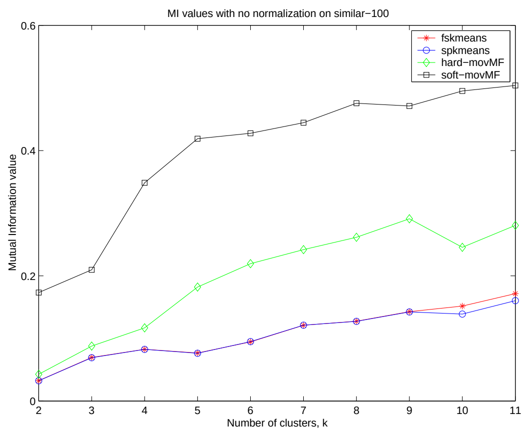
(b) Comparison of MI values for Small-news20.



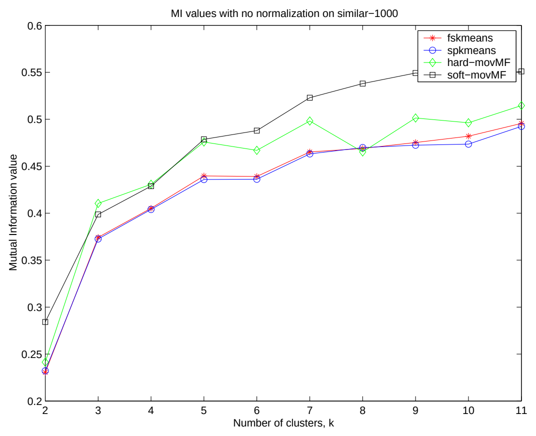
(c) Comparison of MI values for Different-100.



(d) Comparison of MI values for Different-1000.

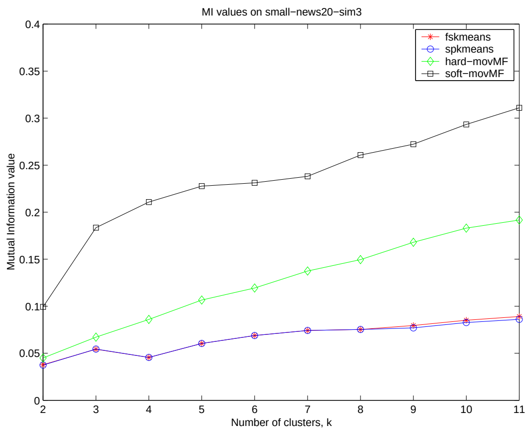


(e) Comparison of MI values for Similar-100.

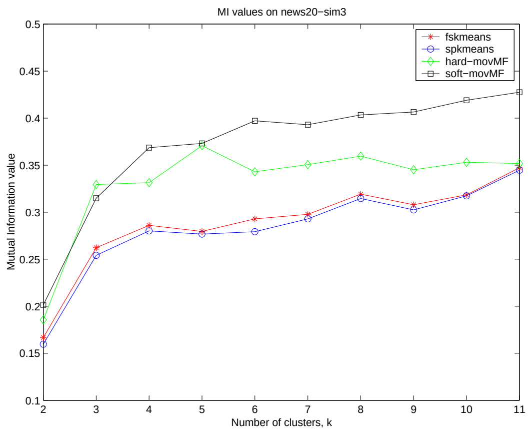


(f) Comparison of MI values for Similar-1000.

Figure 3: Comparison of the algorithms for the CMU Newsgroup and some subsets.



(a) Comparison of MI values for Same-100.



(b) Comparison of MI values for Same-1000.

Figure 4: Comparison of the algorithms for more subsets of CMU Newsgroup data.

6.7 Yeast Gene Expression Data Set

The gene data set that we consider differs from text data in two major aspects. First, the data can have negative values, and second, we do not know the true labels for the data points.

Owing to the absence of true cluster labels for the data points, we evaluate the clusterings by computing certain internal figures of merit. These internal measures have been earlier employed for evaluating clustering of genes (e.g., Sharan and Shamir, 2000). Let $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_n\}$ be the set of data that is clustered into disjoint clusters $\mathcal{X}_1, \dots, \mathcal{X}_k$. Let $\boldsymbol{\mu}_j$ denote the mean vector of the j-th cluster $(1 \le j \le k)$. The homogeneity of the clustering is measured by

$$H_{avg} = \frac{1}{|\mathcal{X}|} \sum_{j=1}^{k} \sum_{\boldsymbol{x} \in \mathcal{X}_j} \frac{\boldsymbol{x}^T \boldsymbol{\mu}_j}{\|\boldsymbol{x}\| \|\boldsymbol{\mu}_j\|}. \tag{6.1}$$

As can easily be seen, a higher homogeneity means that the individual elements of each cluster are quite similar to the cluster representative. We also take note of the minimum similarity

$$H_{min} = \min_{\substack{1 \le j \le k \\ \boldsymbol{x} \in \mathcal{X}_j}} \frac{\boldsymbol{x}^T \boldsymbol{\mu}_j}{\|\boldsymbol{x}\| \|\boldsymbol{\mu}_j\|}. \tag{6.2}$$

Both H_{avg} and H_{min} provide a measure of the intra-cluster similarity. We now define the inter-cluster separation as

$$S_{avg} = \frac{1}{\sum_{i \neq j} |\mathcal{X}_i||\mathcal{X}_j|} \sum_{i \neq j} |\mathcal{X}_i||\mathcal{X}_j| \frac{\boldsymbol{\mu}_i^T \boldsymbol{\mu}_j}{\|\boldsymbol{\mu}_i\| \|\boldsymbol{\mu}_j\|}. \tag{6.3}$$

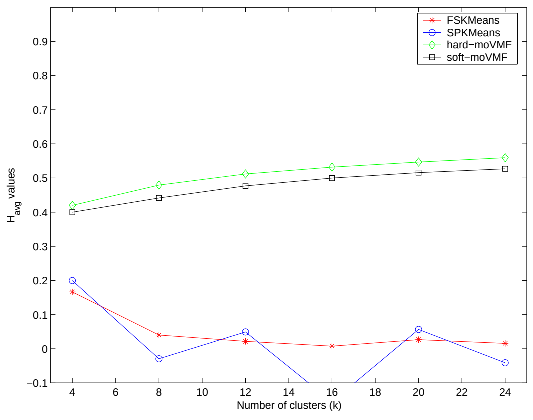
We also take note of the maximum inter-cluster similarity

$$S_{max} = \max_{i \neq j} \frac{\boldsymbol{\mu}_i^T \boldsymbol{\mu}_j}{\|\boldsymbol{\mu}_i\| \|\boldsymbol{\mu}_j\|}. \tag{6.4}$$

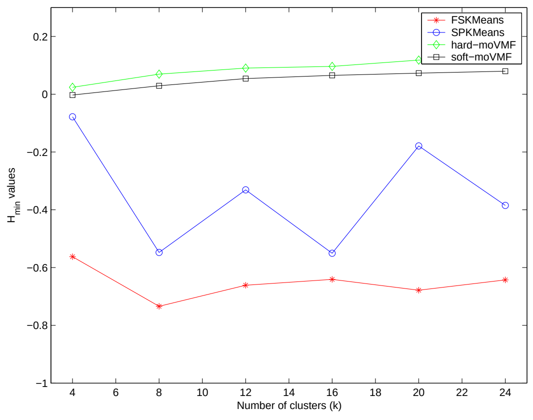
It is easily seen that for a “good” clustering S_{avg} and S_{max} should be low.

Recently, researchers (Segal et al., 2003; Lee et al., 2004) have started looking at supervised methods of evaluating the gene clustering results using public genome databases such as the Kyoto Encyclopedia of Genes and Genomes (KEGG) and the gene ontology (GO). As of now, the evaluation techniques are still evolving and there is no consensus on how to best use the databases. For example, it is becoming clear that a pairwise precision-recall analysis of gene pairs may not be useful since the databases are currently incomplete due to lack of knowledge about all genes. In the recent past, progress has been made in terms of supervised evaluation and online tools such as GoMiner (GoMiner03) have been developed. As future work, we would like to evaluate the performance of our proposed algorithms using such tools.

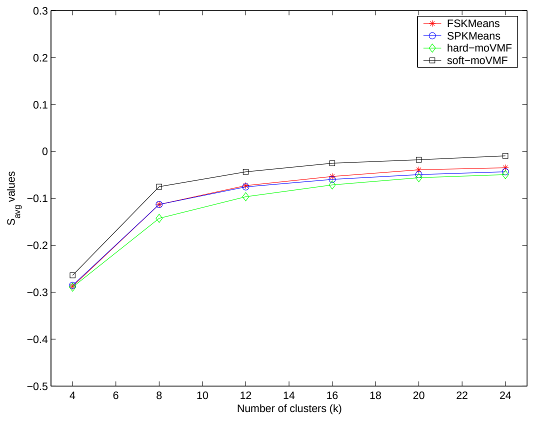
Figure 5 shows the various cluster quality figures of merit as computed for clusters of our gene expression data. A fact that one immediately observes is that `hard-moVMF` consistently performs better than all the other algorithms. This comes as somewhat of a surprise, because in almost all other data sets, `soft-moVMF` performs better (though, of course, the measures of evaluation are different for gene data as compared to the other data sets that we considered). Note that the figures of merit for `soft-moVMF` are computed after "hardening" the clustering results that it produced.



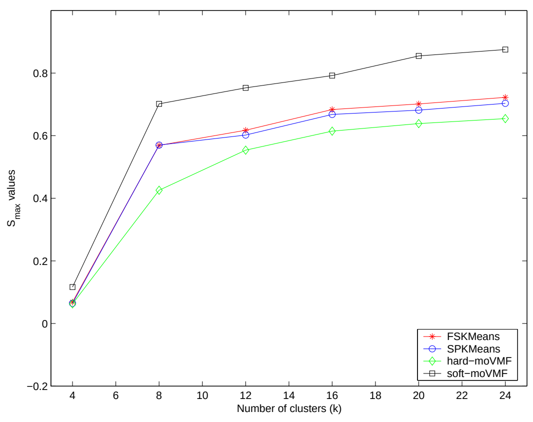
(a) H_{avg} values



(b) H_{min} values



(c) S_{avg} values



(d) S_{max} values

Figure 5: Measures of cluster quality for gene data.

We see from Figure 5(a) that both `hard-moVMF` and `soft-moVMF` yield clusters that are much more homogeneous than those furnished by `fskmeans` and `spkmeans`. The inter-cluster similarities, as measured by S_{avg} and S_{max} are again the lowest for `hard-moVMF`, thereby indicating that `hard-moVMF` gives the best separated clusters of all the four algorithms. Though the inter-cluster similarities do not differ that much between the four algorithms, `soft-moVMF` seems to be forming clusters with higher inter-cluster similarity than other algorithms. We could explain this behavior of `soft-moVMF` by noting that it tends to form overlapping clusters (because of soft-assignments) and those clusters remain closer even after hardening. Since H_{avg} essentially measures the average cosine similarity, we note that using our moVMF based algorithms, we are able to achieve clusters that are more coherent and better separated—a fact that could be attributed to the richer model employed by our algorithms. An inescapable observation is that our vMF based algorithms obtain a better average cosine similarity than `spkmeans`, implying that the richer vMF model allows them to escape the local minima that trap `spkmeans`.

6.8 Running Time

This section shows a brief report of the running time differences between `hard-moVMF` and `soft-moVMF`. Table 8 shows these comparisons. These running time experiments were performed on an AMD Athlon based computer running the Linux operating system. From Table 8 we see that `hard-moVMF`

Clusters	Classic300	Classic3	News20
3	0.39s/11.56s	3.03s/109.87s	10.18s/619.68s
5	0.54s/17.99s	3.59s/163.09s	14.05s/874.13s
10	-	-	18.9s/1512s
20	-	-	29.08s/3368s

Table 8: Running time comparison between `hard-moVMF` and `soft-moVMF`. The times are indicated in the format "`hard-moVMF`/ `soft-moVMF`".

runs much faster than `soft-moVMF`, and this difference becomes even greater when the number of clusters desired becomes higher.

7. Discussion

The mixture of vMF distributions gives a parametric model-based generalization of the widely used cosine similarity measure. As discussed in Section 5.1, the spherical kmeans algorithm that uses cosine similarity arises as a special case of EM on mixture of vMFs when, among other things, the concentration κ of all the distributions is held constant. Interestingly, an alternative and more formal connection can be made from an information geometry viewpoint (Amari, 1995). More precisely, consider a data set that has been sampled following a vMF distribution with a given κ, say $\kappa = 1$. Assuming the Fisher-Information matrix is identity, the Fisher kernel similarity (Jaakkola and Haussler, 1999) corresponding to the vMF distribution is given by

$$\begin{aligned} K(\boldsymbol{x}_i, \boldsymbol{x}_j) &= (\nabla_{\boldsymbol{\mu}} \ln f(\boldsymbol{x}_i|\boldsymbol{\mu}))^T (\nabla_{\boldsymbol{\mu}} \ln f(\boldsymbol{x}_j|\boldsymbol{\mu})) \quad \text{(see (2.1))} \\ &= (\nabla_{\boldsymbol{\mu}} (\boldsymbol{\mu}^T \boldsymbol{x}_i))^T (\nabla_{\boldsymbol{\mu}} (\boldsymbol{\mu}^T \boldsymbol{x}_j)) = \boldsymbol{x}_i^T \boldsymbol{x}_j, \end{aligned}$$

which is exactly the cosine similarity. This provides a theoretical justification for a long-practiced approach in the information retrieval community.

In terms of performance, the magnitude of improvement shown by the `soft-movMF` algorithm for the difficult clustering tasks was surprising, especially since for low-dimensional non-directional data, the improvements using a soft, EM-based `kmeans` or fuzzy kmeans over the standard hard-assignment based versions are often quite minimal. In particular, we were curious regarding a couple of issues: (i) why is `soft-movMF` performing substantially better than `hard-movMF`, even though the final probability values obtained by `soft-movMF` are actually very close to 0 and 1; and (ii) why is `soft-movMF`, which needs to estimate more parameters, doing better even when there are insufficient number of points relative to the dimensionality of the space.

It turns out that both these issues can be understood by taking a closer look at how `soft-moVMF` converges. In all our experiments, we initialized κ to 10, and the initial centroids to small random perturbations of the global centroid. Hence, for `soft-movMF`, the initial posterior membership distributions of the data points are almost uniform and the Shannon entropy of the hidden random variables is very high. The change of this entropy over iterations for the News20 subsets is presented in Figure 6. The behavior is similar for all the other data sets that we studied. Unlike kmeans-based algorithms where most of the relocation happens in the first two or three iterations with only minor adjustments later on, in `soft-movMF` the data points are noncommittal in the first few iterations, and the entropy remains very high (the maximum possible entropy for 3 clusters can be $\log_2 3 = 1.585$). The cluster patterns are discovered only after several iterations, and the entropy drops drastically within a small number of iterations after that. When the algorithm converges, the entropy is practically zero and all points are effectively hard-assigned to their respective clusters. Note that this behavior is strikingly similar to (locally adaptive) annealing approaches where κ can be considered as the inverse of the temperature parameter. The drastic drop in entropy after a few iterations is the typical critical temperature behavior observed in annealing.

As text data has only non-negative features values, all the data points lie in the first orthant of the d-dimensional hypersphere and hence, are naturally very concentrated. The gene-expression data, though spread all over the hypersphere seemed to have some high concentration regions. In either case, the final κ values on convergence are very high, reflecting the concentration in the data, and implying a low final temperature from the annealing perspective. Then, initializing κ to a low value, or equivalently a high temperature, is a good idea because in that case when `soft-movMF` is running, the κ values will keep on increasing over successive iterations to get to its final large values, giving the effect of a decreasing temperature in the process, without any explicit deterministic annealing strategy. Also different mixture components can take different values of κ, as automatically determined by the EM algorithm itself, and need not be controlled by any external heuristic. The cost of the added flexibility in `soft-moVMF` over `spkmeans` is the extra computation in estimating the κ values. Thus the `soft-movMF` algorithm provides a trade-off between modeling power and computational demands, but one that judging from the empirical results, seems quite worthwhile. The `hard-movMF` algorithm, instead of using the more general vMF model, suffers because of hard-assignments from the very beginning. The `fskmeans` and `spkmeans` do not do well for difficult data sets due to their hard assignment scheme as well as their significantly less modeling capabilities.

Finally, a word on model selection, since choosing the number of clusters remains one of the widely debated topics in clustering (McLachlan and Peel, 2000). Banerjee and Langford (2004) have proposed a new objective criterion for evaluation and model-selection for clustering algorithms: how well does the clustering algorithm perform as a prediction algorithm. The prediction

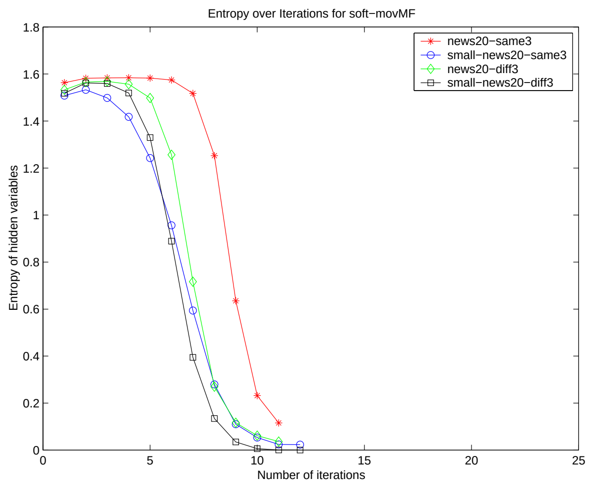


Figure 6: Variation of Entropy of hidden variables with number of Iterations (`soft-movMF`).

accuracy of the clustering is measured by the PAC-MDL bound (Blum and Langford, 2003; Banerjee and Langford, 2004) that upper-bounds the error-rate of predictions on the test-set. The way to use it for model-selection is quite straight-forward: among a range of number of clusters, choose the one that achieves the minimum bound on the test-set error-rate. Experiments on model selection applied to several clustering algorithms were reported by Banerjee and Langford (2004). Interestingly, the movMF-based algorithms almost always obtained the 'right number of clusters'—in this case, the underlying labels in the data set were actually known and the number of labels were considered to be the right number of clusters. It is important to note that this form of model-selection only works in a semi-supervised setting where a little amount of labeled data is available for model selection.

8. Conclusions and Future Work

From the experimental results, it seems that certain high-dimensional data sets, including text and gene-expression data, have properties that match well with the modeling assumptions of the vMF mixture model. This motivates further study of such models. For example, one can consider a hybrid algorithm that employs `soft-moVMF` for the first few (more important) iterations, and then switches to `hard-moVMF` for speed, and measure the speed-quality tradeoff that this hybrid approach provides. Another possible extension would be to consider an online version of the EM-based algorithms as discussed in this paper, developed along the lines of Neal and Hinton (1998). Online algorithms are particularly attractive for dealing with streaming data when memory is limited, and for modeling mildly non-stationary data sources. We could also adapt a local search strategy such as the one in Dhillon et al. (2002a), for incremental EM to yield better local minima for both hard and soft-assignments.

The vMF distribution that we considered in the proposed techniques is one of the simplest parametric distributions for directional data. The iso-density lines of the vMF distribution are circles on the hypersphere, i.e., all points on the surface of the hypersphere at a constant angle from the mean direction. In some applications, more general iso-density contours may be desirable. There are more general models on the unit sphere, such as the Bingham distribution, the Kent distribution, the Watson distribution, the Fisher-Bingham distribution, the Pearson type VII distributions (Shimizu and Iida, 2002; Mardia and Jupp, 2000), etc., that can potentially be more applicable in the general setting. For example, the Fisher-Bingham distributions have added modeling power since there are $O(d^2)$ parameters for each distribution. However, the parameter estimation problem, especially in high-dimensions, can be significantly more difficult for such models, as more parameters need to estimated from the data. One definitely needs substantially more data to get reliable estimates of the parameters. Further, for some cases, e.g., the Kent distribution, it can be difficult to solve the estimation problem in more than 3-dimensions (Peel et al., 2001). Hence these more complex models may not be viable for many high-dimensional problems. Nevertheless, the tradeoff between model complexity (in terms of the number of parameters and their estimation), and sample complexity needs to be studied in more detail in the context of directional data.

Acknowledgments

This research was supported in part by an IBM PhD fellowship to Arindam Banerjee, NSF grant IIS-0307792, ITR-0312471, NSF CAREER Award No. ACI-0093404 and Texas Advanced Research Program grant 003658-0431-2001.

Appendix A. Derivations

For reference, we provide the derivation of Maximum Likelihood Estimates (MLE) for data drawn for a single vMF distribution (Section A.1), and Expectation Minimization update formulae for data drawn from a mixture of k vMF distributions (Section A.2).

A.1 Maximum Likelihood Estimates

In this section we look briefly at maximum likelihood estimates for the parameters of a single vMF distribution. Let $\mathcal{X}$ be a finite set of sample unit vectors drawn independently following $f(\boldsymbol{x}|\boldsymbol{\mu},\kappa)$ (see 2.1), i.e.,

$$\mathcal{X} = \{\boldsymbol{x}_i \in \mathbb{S}^{d-1} \mid \boldsymbol{x}_i \text{ follows } f(\boldsymbol{x}|\boldsymbol{\mu},\kappa) \text{ for } 1 \le i \le n\}.$$

Given $\mathcal{X}$ we want to find maximum likelihood estimates for the parameters $\boldsymbol{\mu}$ and κ of the distribution $f(\boldsymbol{x}|\boldsymbol{\mu},\kappa)$. Assuming the $\boldsymbol{x}_i$ to be independent and identically distributed, we can rewrite the likelihood of $\mathcal{X}$ as

$$P(\mathcal{X}|\boldsymbol{\mu},\kappa) = P(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n|\boldsymbol{\mu},\kappa) = \prod_{i=1}^{n} f(\boldsymbol{x}_i|\boldsymbol{\mu},\kappa) = \prod_{i=1}^{n} c_d(\kappa) e^{\kappa \boldsymbol{\mu}^T \boldsymbol{x}_i}. \tag{A.1}$$

Taking the logarithm on both sides of (A.1) we obtain

$$\ln P(\mathcal{X}|\boldsymbol{\mu},\kappa) = n \ln c_d(\kappa) + \kappa \boldsymbol{\mu}^T \boldsymbol{r}, \tag{A.2}$$

where $\boldsymbol{r} = \sum_i \boldsymbol{x}_i$. To obtain the maximum likelihood estimates of $\boldsymbol{\mu}$ and κ, we have to maximize (A.2), subject to the constraints $\boldsymbol{\mu}^T\boldsymbol{\mu} = 1$ and $\kappa \ge 0$. Introducing a Lagrange multiplier λ, the Lagrangian of the objective function is given by[5]

$$L(\boldsymbol{\mu},\kappa,\lambda;\mathcal{X}) = n\ln c_d(\kappa) + \kappa\boldsymbol{\mu}^T\boldsymbol{r} + \lambda(1-\boldsymbol{\mu}^T\boldsymbol{\mu}). \tag{A.3}$$

Differentiating the Lagrangian (A.3) with respect to $\boldsymbol{\mu}$, λ and κ and setting the derivatives to zero, we get the following equations that the parameter estimates $\hat{\boldsymbol{\mu}}$, $\hat{\lambda}$ and $\hat{\kappa}$ must satisfy:

$$\hat{\boldsymbol{\mu}} = \frac{\hat{\kappa}}{2\hat{\lambda}}\boldsymbol{r}, \tag{A.4a}$$

$$\hat{\boldsymbol{\mu}}^T\hat{\boldsymbol{\mu}} = 1, \tag{A.4b}$$

$$\frac{nc_d'(\hat{\kappa})}{c_d(\hat{\kappa})} = -\hat{\boldsymbol{\mu}}^T\boldsymbol{r}. \tag{A.4c}$$

Substituting (A.4a) in (A.4b) gives us

$$\hat{\lambda} = \frac{\hat{\kappa}}{2}\|\boldsymbol{r}\|, \tag{A.5}$$

$$\text{and} \quad \hat{\boldsymbol{\mu}} = \frac{\boldsymbol{r}}{\|\boldsymbol{r}\|} = \frac{\sum_{i=1}^n \boldsymbol{x}_i}{\|\sum_{i=1}^n \boldsymbol{x}_i\|}. \tag{A.6}$$

Substituting $\hat{\boldsymbol{\mu}}$ from (A.6) in (A.4c) we obtain

$$\frac{c_d'(\hat{\kappa})}{c_d(\hat{\kappa})} = -\frac{\|\boldsymbol{r}\|}{n} = -\bar{r}. \tag{A.7}$$

For brevity, let us write $s = d/2-1$ and $\xi = (2\pi)^{s+1}$; on differentiating (2.2) with respect to κ, we obtain

$$c_d'(\kappa) = \frac{s\kappa^{s-1}}{\xi I_s(\kappa)} - \frac{\kappa^s I_s'(\kappa)}{\xi I_s^2(\kappa)}.$$

The right-hand-side simplifies to

$$\frac{\kappa^s}{\xi I_s(\kappa)}\left(\frac{s}{\kappa} - \frac{I_s'(\kappa)}{I_s(\kappa)}\right) = c_d(\kappa)\left(\frac{s}{\kappa} - \frac{I_s'(\kappa)}{I_s(\kappa)}\right).$$

Using the following well known recurrence relation (see Abramowitz and Stegun (1974, Sec. 9.6.26)),

$$\kappa I_{s+1}(\kappa) = \kappa I_s'(\kappa) - sI_s(\kappa),$$

we find that

$$\frac{-c_d'(\kappa)}{c_d(\kappa)} = \frac{I_{s+1}(\kappa)}{I_s(\kappa)} = \frac{I_{d/2}(\kappa)}{I_{d/2-1}(\kappa)}.$$

Thus we can obtain the estimate $\hat{\kappa}$ by solving

$$A_d(\hat{\kappa}) = \bar{r}, \tag{A.8}$$

where $A_d(\kappa) = \frac{I_{d/2}(\kappa)}{I_{d/2-1}(\kappa)}$ and $\bar{r} = \|\boldsymbol{r}\|/n$. Since $A_d(\kappa)$ is a ratio of Bessel functions, it is not possible to obtain a closed form expression for A_d^{-1}. We have to take recourse to numerical or asymptotic methods to obtain an approximation for κ.

5. Strictly speaking, we should introduce the inequality constraint in the Lagrangian for κ, and work with the necessary KKT conditions. However if $\kappa = 0$ then $f(\boldsymbol{x}|\boldsymbol{\mu},\kappa)$ is the uniform distribution on the sphere, and if $\kappa > 0$ then the multiplier for the inequality constraint has to be zero by the KKT condition, so the Lagrangian in (A.3) is adequate.

A.2 Expectation Maximization (EM)

Suppose the posterior distribution, $p(h|\boldsymbol{x}_i,\Theta)$, of the hidden variables $\mathcal{Z}|(\mathcal{X},\Theta)$ is known. Unless otherwise specified, henceforth all expectations will be taken over the distribution of the (set of) random variable(s) $\mathcal{Z}|(\mathcal{X},\Theta)$. Expectation of the complete data log-likelihood (see 3.2) over the given posterior distribution p can be written as

$$\begin{aligned} E_p[\ln P(\mathcal{X},\mathcal{Z}|\Theta)] &= \sum_{i=1}^{n} E_p[\ln(\alpha_{\boldsymbol{z}_i} f_{\boldsymbol{z}_i}(\boldsymbol{x}_i|\theta_{\boldsymbol{z}_i}))] \\ &= \sum_{i=1}^{n}\sum_{h=1}^{k} \ln(\alpha_h f_h(\boldsymbol{x}_i|\theta_h))\ p(h|\boldsymbol{x}_i,\Theta) \qquad \text{(A.9)} \\ &= \sum_{h=1}^{k}\sum_{i=1}^{n}(\ln\alpha_h)\ p(h|\boldsymbol{x}_i,\Theta) + \sum_{h=1}^{k}\sum_{i=1}^{n}(\ln f_h(\boldsymbol{x}_i|\theta_h))\ p(h|\boldsymbol{x}_i,\Theta). \end{aligned}$$

In the parameter estimation or M-step, Θ is re-estimated so that the above expression is maximized. Note that for maximizing this expectation we can separately maximize the terms containing α_h and θ_h as they are unrelated (observe that $p(h|\boldsymbol{x}_i,\Theta)$ is fixed).

To maximize the expectation with respect to each α_h we introduce a Lagrangian multiplier λ corresponding to the constraint $\sum_{h=1}^{k}\alpha_h = 1$. We form the Lagrangian, and take partial derivatives with respect to each α_h obtaining

$$\sum_{i=1}^{n} p(h|\boldsymbol{x}_i,\Theta) = -\lambda\hat{\alpha}_h. \qquad \text{(A.10)}$$

On summing both sides of (A.10) over all h we find that $\lambda = -n$, hence

$$\hat{\alpha}_h = \frac{1}{n}\sum_{i=1}^{n} p(h|\boldsymbol{x}_i,\Theta). \qquad \text{(A.11)}$$

Next we concentrate on terms containing $\theta_h = (\boldsymbol{\mu}_h,\kappa_h)$ under the constraints $\boldsymbol{\mu}_h^T\boldsymbol{\mu}_h = 1$ and $\kappa_h \geq 0$ for $1 \leq h \leq k$. Let λ_h be the Lagrange multiplier corresponding to each equality constraint (see footnote on page 1374). The Lagrangian is given by

$$\begin{aligned} L(\{\boldsymbol{\mu}_h,\kappa_h,\lambda_h\}_{h=1}^{k}) &= \sum_{h=1}^{k}\sum_{i=1}^{n}(\ln f_h(\boldsymbol{x}_i|\theta_h))\ p(h|\boldsymbol{x}_i,\Theta) + \sum_{h=1}^{k}\lambda_h\left(1-\boldsymbol{\mu}_h^T\boldsymbol{\mu}_h\right) \\ &= \sum_{h=1}^{k}\left[\sum_{i=1}^{n}(\ln c_d(\kappa_h))\ p(h|\boldsymbol{x}_i,\Theta) + \sum_{i=1}^{n}\kappa_h\boldsymbol{\mu}_h^T\boldsymbol{x}_i\ p(h|\boldsymbol{x}_i,\Theta) + \lambda_h(1-\boldsymbol{\mu}_h^T\boldsymbol{\mu}_h)\right]. \end{aligned} \qquad \text{(A.12)}$$

Taking partial derivatives of (A.12) with respect to $\{\boldsymbol{\mu}_h,\lambda_h,\kappa_h\}_{h=1}^{k}$ and setting them to zero, for each h we get:

$$\boldsymbol{\mu}_h = \frac{\kappa_h}{2\lambda_h}\sum_{i=1}^{n}\boldsymbol{x}_i p(h|\boldsymbol{x}_i,\Theta), \qquad \text{(A.13a)}$$

$$\boldsymbol{\mu}_h^T\boldsymbol{\mu}_h = 1, \qquad \text{(A.13b)}$$

$$\frac{c_d'(\kappa_h)}{c_d(\kappa_h)}\sum_{i=1}^{n} p(h|\boldsymbol{x}_i,\Theta) = -\boldsymbol{\mu}_h^T\sum_{i=1}^{n}\boldsymbol{x}_i p(h|\boldsymbol{x}_i,\Theta). \qquad \text{(A.13c)}$$

Using (A.13a) and (A.13b) we get

$$\lambda_h = \frac{\kappa_h}{2}\left\| \sum_{i=1}^{n} \boldsymbol{x}_i p(h|\boldsymbol{x}_i,\Theta)\right\|,$$
$$\boldsymbol{\mu}_h = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i p(h|\boldsymbol{x}_i,\Theta)}{\|\sum_{i=1}^{n} \boldsymbol{x}_i p(h|\boldsymbol{x}_i,\Theta)\|}. \tag{A.14}$$

Substituting (A.14) in (A.13c) gives us

$$\frac{c_d'(\kappa_h)}{c_d(\kappa_h)} = -\frac{\|\sum_{i=1}^{n} \boldsymbol{x}_i p(h|\boldsymbol{x}_i,\Theta)\|}{\sum_{i=1}^{n} p(h|\boldsymbol{x}_i,\Theta)}, \tag{A.15}$$

which can be written as

$$A_d(\kappa_h) = \frac{\|\sum_{i=1}^{n} \boldsymbol{x}_i p(h|\boldsymbol{x}_i,\Theta)\|}{\sum_{i=1}^{n} p(h|\boldsymbol{x}_i,\Theta)}, \tag{A.16}$$

where $A_d(\kappa) = \frac{I_{d/2}(\kappa)}{I_{d/2-1}(\kappa)}$. Note that (A.14) and (A.16) are intuitive generalizations of (A.6) and (A.8) respectively.

A.3 Experimental Study of the Approximation

In this section we provide a brief experimental study to assess the quality of our approximation of the concentration parameter κ. Recall that our approximation (4.4) attempts to solve the implicit non-linear equation

$$\frac{I_{d/2}(\kappa)}{I_{d/2-1}(\kappa)} = \bar{r}. \tag{A.17}$$

We previously mentioned that for large values of $\bar{r}$ ($\bar{r}$ close to 1), approximation (4.1) is reasonable; for small values of $\bar{r}$ (usually for $\bar{r} < 0.2$) estimate (4.2) is quite good; Eqn. (4.4) yields good approximations for most values of $\bar{r}$.

A particular value of $\bar{r}$ may correspond to many different combinations of κ and d values. Thus, to assess the quality of various approximations, we need to evaluate their performance across the (κ, d) plane. However, such an assessment is difficult to illustrate through 2-dimensional plots. To supplement Table 1, which showed how the three approximations behave on a sampling of points from the (κ, d) plane, in this section we present experimental results on some slices of this plane, where we either keep d fixed and vary κ, or we keep κ fixed and vary d. For all our evaluations, the $\bar{r}$ values were computed using (A.17).

We begin by holding d fixed at 1000, and allow κ to vary from 10 to 5010. Figure 7 shows the values of computed $\hat{\kappa}$ (estimation of κ) using the three approximations. From this figure one can see that (4.1) overestimates the true κ, while (4.2) underestimates it. However, our approximation (4.4) is very close to the true κ values.

Next we illustrate the quality of approximation when κ is held fixed and d is allowed to vary. Figure 8 illustrates how the various approximations behave as the dimensionality d is varied from $d = 4$ till $d = 1454$. The concentration parameter κ was set at 500 for this experiment. We see that (4.2) catches up with the true value of κ after approximately $d \geq 2\kappa$ (because the associated $\bar{r}$ values become small), whereas (4.4) remains accurate throughout.

Since all the approximations depend on $\bar{r}$ (which implicitly depends on κ and d), it is illustrative to also plot the approximation errors as $\bar{r}$ is allowed to vary. Figure 9 shows how the three

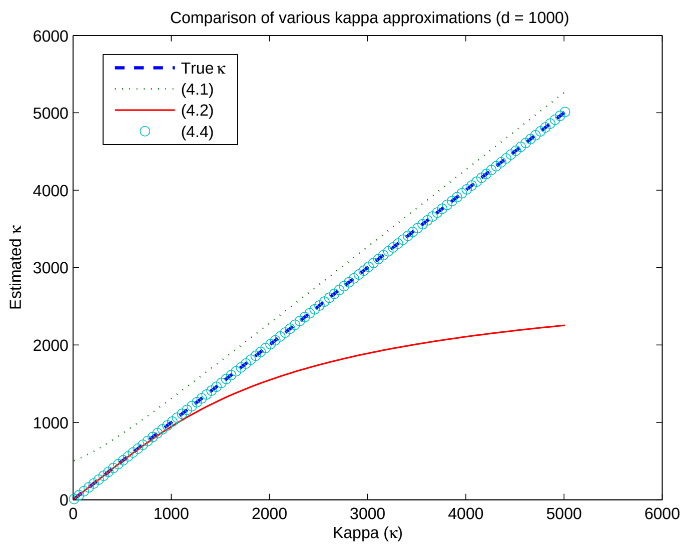


Figure 7: Comparison of true and approximated κ values, with $d = 1000$

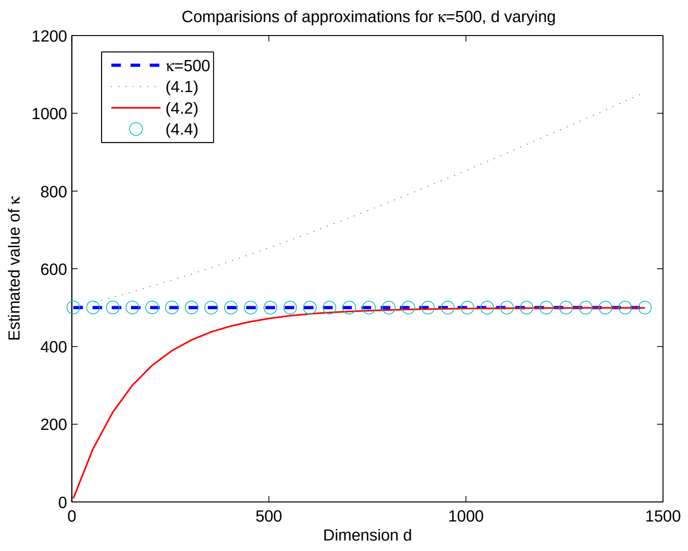


Figure 8: Comparison of approximations for varying d, $\kappa = 500$.

approximations perform as $\bar{r}$ ranges from 0.05 to 0.95. Let $f(d,\bar{r})$, $g(d,\bar{r})$, and $h(d,\bar{r})$ represent the approximations to κ using (4.1), (4.2) and (4.4), respectively. Figure 9 displays $|A_d(f(d,\bar{r})) - \bar{r}|$, $|A_d(g(d,\bar{r})) - \bar{r}|$, and $|A_d(h(d,\bar{r})) - \bar{r}|$ for the varying $\bar{r}$ values. Note that the y-axis is on a log-scale to appreciate the differences between the three approximations. We see that up to $\bar{r} \approx 0.18$ (dashed line on the plot), the approximation yielded by (4.2) has lower error. Thereafter, approximation (4.4) becomes better.

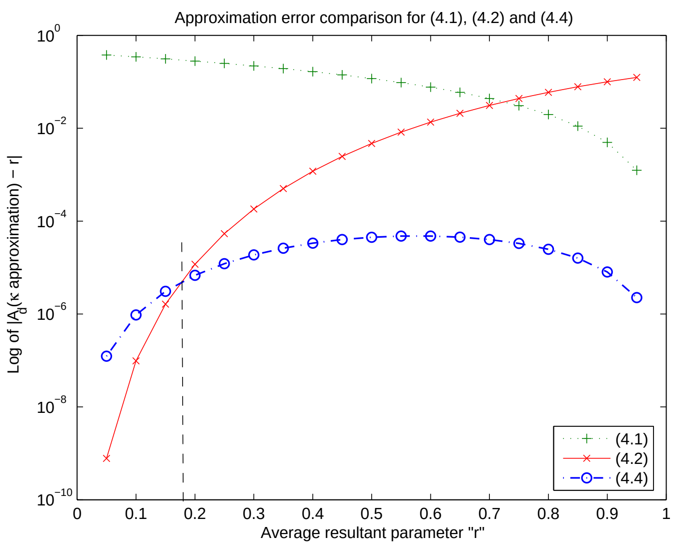


Figure 9: Comparison of approximations for varying $\bar{r}$ (with $d = 1000$)

References

M. Abramowitz and I. A. Stegun, editors. *Handbook of Mathematical Functions*. Dover Publ. Inc., New York, 1974.

S. I. Amari. Information geometry of the EM and em algorithms for neural networks. *Neural Networks*, 8(9):1379–1408, 1995.

A. Banerjee and J. Ghosh. Frequency sensitive competitive learning for clustering on high-dimensional hyperspheres. In *Proceedings International Joint Conference on Neural Networks*, pages 1590–1595, May 2002.

A. Banerjee and J. Ghosh. Frequency Sensitive Competitive Learning for Scalable Balanced Clustering on High-dimensional Hyperspheres. *IEEE Transactions on Neural Networks*, 15(3):702–719, May 2004.

A. Banerjee and J. Langford. An objective evaluation criterion for clustering. In *Proc. 10th International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 515–520, 2004.

A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with Bregman divergences. In *Proc. of 4th SIAM International Conference on Data Mining*, pages 234–245, 2004.

J. Bilmes. A Gentle Tutorial on the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models. Technical Report ICSI-TR-97-021, University of Berkeley, 1997.

A. Blum and J. Langford. PAC-MDL bounds. In *Proc. 16th Annual Conference on Learning Theory (COLT)*, 2003.

P. S. Bradley, K. P. Bennett, and A. Demiriz. Constrained k-means clustering. Technical report, Microsoft Research, May 2000.

D. Coleman, X. Dong, J. Hardin, D. Rocke, and D. Woodruf. Some computational issues in cluster analysis with no apriori metric. *Computional Statistics and Data Analysis*, 31:1–11, 1999.

M. Collins. The EM algorithm. In fulfillment of Written Preliminary Exam II requirement, September 1997.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 1991.

S. Dasgupta. Learning mixtures of Gaussians. In *IEEE Symposium on Foundations of Computer Science*, 1999.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.

I. S. Dhillon, J. Fan, and Y. Guan. Efficient clustering of very large document collections. In V. Kumar R. Grossman, C. Kamath and R. Namburu, editors, *Data Mining for Scientific and Engineering Applications*. Kluwer Academic Publishers, 2001.

I. S. Dhillon, Y. Guan, and J. Kogan. Iterative clustering of high dimensional text data augmented by local search. In *Proceedings of The 2002 IEEE International Conference on Data Mining*, 2002a.

I. S. Dhillon, E. M. Marcotte, and U. Roshan. Diametrical clustering for identifying anti-correlated gene clusters. Technical Report TR 2002-49, Department of Computer Sciences, University of Texas, September 2002b.

I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1):143–175, 2001.

I. S. Dhillon and S. Sra. Modeling data using directional distributions. Technical Report TR-03-06, Department of Computer Sciences, University of Texas at Austin, Austin, TX, 2003.

B. E. Dom. An information-theoretic external cluster-validity measure. Technical Report RJ 10219, IBM Research Report, 2001.

M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc. National Academy of Science*, 95:14863–14868, 1998.

N. I. Fisher. *Statistical Analysis of Circular Data*. Cambridge University Press, 1996.

J. Ghosh. Scalable clustering methods for data mining. In Nong Ye, editor, *Handbook of Data Mining*, pages 247–277. Lawrence Erlbaum, 2003.

GoMiner03. http://discover.nci.nih.gov/gominer/.

T. R. Hughes, M. J. Marton, A. R. Jones, C. J. Roberts, R. Stoughton, C. D. Armour, H. A. Bennett, E. Coffey, H. Dai, D. D. Shoemaker, D. Gachotte, K. Chakraburtty, J. Simon, M. Bard, and S. H. Friend. Functional discovery via a compendium of expression profiles. *Cell*, 102:109–126, 2000.

P. Indyk. A sublinear-time approximation scheme for clustering in metric spaces. In *40th Symposium on Foundations of Computer Science*, 1999.

T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In M. S. Kearns, S. A. Solla, and D. D. Cohn, editors, *Advances in Neural Information Processing Systems*, volume 11, pages 487–493. MIT Press, 1999.

A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, New Jersey, 1988.

R. Kannan, S. Vempala, and A. Vetta. On clusterings—good, bad and spectral. In *41st Annual IEEE Symposium Foundations of Computer Science*, pages 367–377, 2000.

M. Kearns, Y. Mansour, and A. Ng. An information-theoretic analysis of hard and soft assignment methods for clustering. In *13th Annual Conf. on Uncertainty in Artificial Intelligence (UAI97)*, 1997.

I. Lee, S. V. Date, A. T. Adai, and E. M. Marcotte. A probabilistic functional network of yeast genes. *Science*, 306(5701):1555–1558, 2004.

K. V. Mardia. *Statistical Distributions in Scientific Work*, volume 3, chapter "Characteristics of directional distributions", pages 365–385. Reidel, Dordrecht, 1975.

K. V. Mardia and P. Jupp. *Directional Statistics*. John Wiley and Sons Ltd., 2nd edition, 2000.

G. J. McLachlan. The classification and mixture maximum likelihood approaches to cluster analysis. In P. R. Krishnaiah and L. N. Kanal, editors, *Handbook of Statistics*, volume 2, pages 199–208. North-Holland, 1982.

G. J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley-Interscience, 1997.

G. J. McLachlan and D. Peel. *Finite Mixture Models*. Wiley series in Probability and Mathematical Statistics: Applied Probability and Statistics Section. John Wiley & Sons, 2000.

M. Meilă. Comparing clusterings by the variation of information. In *COLT*, 2003.

J. A. Mooney, P. J. Helms, and I. T. Jolliffe. Fitting mixtures of von Mises distributions: a case study involving sudden infant death syndrome. *Computational Statistics & Data Analysis*, 41: 505–513, 2003.

R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. MIT Press, 1998.

20 Newsgroups. http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.html.

K. Nigam, A. K. Mccallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134, 2000.

D. Peel, W. J. Whiten, and G. J. McLachlan. Fitting mixtures of Kent distributions to aid in joint set identification. *Journal of American Statistical Association*, 96:56–63, 2001.

J. H. Piater. *Visual Feature Learning*. PhD thesis, University of Massachussets, June 2001.

C. R. Rao. *Linear Statistical Inference and its Applications*. Wiley, New York, 2nd edition, 1973.

E. Rasmussen. Clustering algorithms. In W. Frakes and R. Baeza-Yates, editors, *Information Retrieval: Data Structures and Algorithms*, pages 419–442. Prentice Hall, New Jersey, 1992.

G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 4(5):513–523, 1988.

G. Salton and M. J. McGill. *Introduction to Modern Retrieval*. McGraw-Hill Book Company, 1983.

B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *World Wide Web*, 10, pages 285–295, 2001.

B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, 2001.

E. Segal, A. Battle, and D. Koller. Decomposing gene expression into cellular processes. In *Proc. of 8th Pacific Symposium on Biocomputing (PSB)*, 2003.

R. Sharan and R. Shamir. CLICK: A clustering algorithm with applications to gene expression analysis. In *Proceedings of 8th International Conference on Intelligent Systems for Molecular Biology (ISMB)*, pages 307–316. AAAI Press, 2000.

K. Shimizu and K. Iida. Pearson type VII distributions on spheres. *Communications in Statistics: Theory & Methods*, 31(4):513–526, 2002.

J. Sinkkonen and S. Kaski. Clustering based on conditional distributions in an auxiliary space. *Neural Computation*, 14:217–239, 2001.

P. Smyth. Clustering sequences with hidden Markov models. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing*, volume 9, pages 648–654. MIT Press, 1997.

M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, 2000.

A. Strehl, J. Ghosh, and R. Mooney. Impact of similarity measures on web-page clustering. In *Proc 7th Natl Conf on Artificial Intelligence : Workshop of AI for Web Search (AAAI 2000)*, pages 58–64. AAAI, July 2000.

C. S. Wallace and D. L. Dowe. MML clustering of multi-state, Poisson, von Mises circular and Gaussian distributions. *Statistics and Computing*, 10(1):73–83, January 2000.

G. N. Watson. *A treatise on the theory of Bessel functions*. Cambridge University Press, 2nd edition, 1995.

A. T. A. Wood. Simulation of the von-Mises Distribution. *Communications of Statistics, Simulation and Computation*, 23:157–164, 1994.

Y. Zhao and G. Karypis. Empirical and theoretical comparisons of selected criterion functions for document clustering. *Machine Learning*, 55(3):311–331, June 2004.

S. Zhong and J. Ghosh. A Unified Framework for Model-based Clustering. *Journal of Machine Learning Research*, 4:1001–1037, November 2003a.

S. Zhong and J. Ghosh. A comparative study of generative models for document clustering. In *Workshop on Clustering High Dimensional Data : Third SIAM Conference on Data Mining*, April 2003b.