Chapter 1

Text Clustering with Mixture of von Mises-Fisher Distributions

Arindam Banerjee

University of Minnesota, Twin Cities

Inderjit Dhillon

University of Texas at Austin

Joydeep Ghosh

University of Texas at Austin

Suvrit Sra

Max-Planck Institute for Biological Cybernetics, Tübingen

1.1 Introduction

There is a long-standing folklore in the information retrieval community that a vector space representation of text data has directional properties, i.e., the direction of the vector is much more important than its magnitude. This belief has led to practices such as using the cosine between two vectors for measuring similarity between the corresponding text documents, and to the scaling of vectors to unit L_2 norm [41, 40, 20].

In this chapter, we describe a probabilistic generative model [44, 25] based on directional distributions [30] for modeling text data.[1] Specifically, we suggest that a set of text documents that form multiple topics can be well modeled by a mixture of von Mises-Fisher (vMF) distributions, with each component corresponding to a topic. Generative models often provide greater insights into the anatomy of the data as compared to discriminative approaches. Moreover, domain knowledge can be easily incorporated into generative models, for example, in this chapter the directional nature of the data is reflected in our choice of vMF distributions as the mixture components.

We derive two clustering algorithms based on Expectation Maximization (EM) for estimating the parameters of the mixture model from first principles. Our algorithms involve estimating a *concentration* parameter, κ, for each component of the mixture model. 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 spherical kmeans (`spkmeans`) [20] as a model based on a mixture of identity covariance Gaussians has to classical `kmeans` that uses squared Euclidean distances [9]. 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.

While this chapter focuses on text analysis, we note that many other important domains such as bioinformatics and collaborative filtering involve directional data as well. Thus, the scope and applications of the approaches taken in this chapter are much broader and not limited to text alone.

The remainder of the chapter is organized as follows. In section 1.2, we discuss related work on mixture models, text clustering, and vMF distributions. We review the multivariate vMF distribution in Section 1.3. In Section 1.4 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 1.5 highlights our new method of approximating κ and also presents a mathematical analysis of hard assignments. Sections 1.4 and 1.5 form the basis for two clustering algorithms using soft and hard-assignments, respectively, and these algorithms are described in Section 1.6. Detailed experimental results and comparisons with other algorithms are offered in Section 1.7. A discussion on the behavior of our algorithms and a connection with simulated annealing follows in Section 1.8, and we conclude in Section 1.9.

Notation. Bold faced variables, e.g., $\mathbf{x}, \boldsymbol{\mu}$ represent vectors; the norm $\|\cdot\|$ denotes the L_2 norm; sets are represented by script-style upper-case letters,

[1]This chapter treats L_2 normalized data and directional data as synonymous.

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.

1.2 Related Work

There has been an enormous amount of work on clustering a wide variety of datasets across multiple disciplines over the past fifty years [26]. The methods presented in this chapter are tailored for high-dimensional data with directional characteristics, rather than for arbitrary datasets. 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 [48, 54]. For example, hierarchical agglomerative methods based on cosine, Jaccard or Dice coefficients were dominant for text clustering till the mid-1990s [39]. Over the past few years several new approaches, ranging from spectral partitioning [27, 54], to the use of generative models from the exponential family, e.g., mixture of multinomials or Bernoulli distributions [35] etc., have emerged. A fairly extensive list of references on generative approaches to text clustering can be found in [55].

Of particular relevance to this work is the `spkmeans` algorithm [20], 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 [49]. It also provides better characterization of clusters in terms of their top representative or discriminative terms.

The vMF distribution is known in the literature on directional statistics [30], and the maximum likelihood estimates (MLE) of the parameters have been given for a single distribution. Recently Piater [37] obtained parameter estimates for a mixture for circular, i.e., 2-dimensional vMFs. In an Appendix to his thesis, Piater 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. [33] use a mixture of two circular von Mises distributions to estimate the parameters using a quasi-Newton procedure. Wallace and Dowe [51] perform mixture modeling for circular von Mises distributions and have produced a software called Snob that implements their ideas. McLachlan and Peel [31] discuss mixture analysis of directional data and mention the possibility of using Fisher distributions (3-dimensional vMFs), but instead use 3-dimensional Kent distributions [30]. 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, [31] also draws attention to the difficulty of parameter estimation even for 3-D vMFs.

The connection between a generative model involving vMF distributions with constant κ and the `spkmeans` algorithm was first observed by [6]. A variant that could adapt in an on-line fashion leading to balanced clustering solutions was developed by [7]. 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 [43]. 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 movMF model that constrains the concentration κ to be the same for all mixture components during any iteration was presented by [56]. Even with this restriction, this model was superior to both hard and soft versions of multivariate Bernoulli and multinomial models. In recent years, the movMF model has been successfully applied to text mining and anomaly detection applications for the NASA Aviation Safety Reporting System (ASRS) [47, 46].

Recently, [10] discussed the modeling of high dimensional directional data using mixtures of Watson distributions, mainly to handle axial symmetries in the data. The authors of [10] followed the parameter estimation techniques developed in this chapter to obtain numerical estimates for the concentration parameter κ for Watson distributions. Additionally, alternate parameter estimates along with a connection of mixture of Watson based models to *diametric clustering* [19] were developed in [45]. For text data, mixtures of Watson distributions usually perform inferior to moVMF based models, though for gene expression data they could be potentially better.

1.3 Preliminaries

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

1.3.1 The von Mises-Fisher (vMF) Distribution

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

$$f(\mathbf{x}|\mu,\kappa) = c_d(\kappa)e^{\kappa\mu^T\mathbf{x}} \ , \tag{1.1}$$

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

$$c_d(\kappa) = \frac{\kappa^{d/2-1}}{(2\pi)^{d/2}I_{d/2-1}(\kappa)} \ , \tag{1.2}$$

where $I_p(\cdot)$ represents the modified Bessel function of the first kind and order p, and is defined as [1]

$$I_p(\kappa) = \sum_{k\geq 0} \frac{1}{\Gamma(p+k+1)k!}\left(\frac{\kappa}{2}\right)^{2k+p} \ ,$$

where $\Gamma(\cdot)$ is the well-known Gamma function.

The density $f(\mathbf{x}|\mu,\kappa)$ is parameterized by the mean direction μ, and the *concentration* parameter κ, so-called because it characterizes how strongly the unit vectors drawn according to $f(\mathbf{x}|\mu,\kappa)$ are concentrated about the mean direction μ. Larger values of κ imply stronger concentration about the mean direction. In particular when $\kappa = 0$, $f(\mathbf{x}|\mu,\kappa)$ reduces to the uniform density on $\mathbb{S}^{d-1}$, and as $\kappa \to \infty$, $f(\mathbf{x}|\mu,\kappa)$ tends to a point density. The interested reader is referred to [30], [24] or [21] 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 multivariate Gaussian distribution for data in $\mathbb{R}^d$. For example, the maximum entropy density on $\mathbb{S}^{d-1}$ subject to the constraint that $E[\mathbf{x}]$ is fixed is a vMF density (see [38, pp. 172–174] and [29] for details).

1.3.2 Maximum Likelihood Estimates

In this section we look briefly at maximum likelihood estimates for the parameters of a single vMF distribution. The detailed derivations can be found in [5]. Let $\mathcal{X}$ be a finite set of sample unit vectors drawn independently following $f(\mathbf{x}|\mu,\kappa)$ (1.1), i.e.,

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

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

$$\ln P(\mathcal{X}|\mu,\kappa) = n\ln c_d(\kappa) + \kappa\mu^T\mathbf{r}, \tag{1.3}$$

where $\mathbf{r} = \sum_i \mathbf{x}_i$. To obtain the maximum likelihood estimates of μ and κ, we have to maximize (1.3) subject to the constraints $\mu^T\mu = 1$ and $\kappa \geq 0$. A simple calculation [5] shows that the MLE solutions $\hat{\mu}$ and $\hat{\kappa}$ may be obtained from the following equations:

$$\hat{\mu} = \frac{\mathbf{r}}{\|\mathbf{r}\|} = \frac{\sum_{i=1}^{n} \mathbf{x}_i}{\|\sum_{i=1}^{n} \mathbf{x}_i\|}, \tag{1.4}$$

$$\text{and} \qquad \frac{I_{d/2}(\hat{\kappa})}{I_{d/2-1}(\hat{\kappa})} = \frac{\|\mathbf{r}\|}{n} = \bar{r}. \tag{1.5}$$

Since computing $\hat{\kappa}$ involves an implicit equation (1.5) that is a ratio of Bessel functions, it is not possible to obtain an analytic solution, and we have to resort to numerical or asymptotic methods to obtain an approximation (see Section 1.5).

1.4 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, and obtain the update equations for estimating the mixture-density parameters from a given dataset using the Expectation Maximization (EM) framework. Let $f_h(\mathbf{x}|\theta_h)$ denote a vMF distribution with parameters $\theta_h = (\mu_h, \kappa_h)$ for $1 \leq h \leq k$. Then a mixture of these k vMF distributions has a density given by

$$f(\mathbf{x}|\Theta) = \sum_{h=1}^{k} \alpha_h f_h(\mathbf{x}|\theta_h), \tag{1.6}$$

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 $f_h(\mathbf{x}|\theta_h)$. Let $\mathcal{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ be a dataset of n independently sampled points that follow (1.6). Let $\mathcal{Z} = \{\mathbf{z}_1, \cdots, \mathbf{z}_n\}$ be the corresponding set of hidden random variables that indicate the particular vMF distribution from which the points are sampled. In particular, $\mathbf{z}_i = h$ if $\mathbf{x}_i$ is sampled from $f_h(\mathbf{x}|\theta_h)$. Assuming that the values in the set $\mathcal{Z}$ are known, the log-likelihood of the observed data is given by

$$\ln P(\mathcal{X}, \mathcal{Z}|\Theta) = \sum_{i=1}^{n} \ln\left(\alpha_{\mathbf{z}_i} f_{\mathbf{z}_i}(\mathbf{x}_i|\theta_{\mathbf{z}_i})\right). \tag{1.7}$$

Obtaining maximum likelihood estimates for the parameters would have been easy were the $\mathbf{z}_i$ truly known. Unfortunately that is not the case, and (1.7) 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 (1.7) with the constraints $\mu_h^T\mu_h = 1$ and $\kappa_h \geq 0$, we obtain

$$\alpha_h = \frac{1}{n}\sum_{i=1}^{n} p(h|\mathbf{x}_i,\Theta), \tag{1.8}$$

$$\mathbf{r}_h = \sum_{i=1}^{n} \mathbf{x}_i p(h|\mathbf{x}_i,\Theta), \tag{1.9}$$

$$\hat{\mu}_h = \frac{\mathbf{r}_h}{\|\mathbf{r}_h\|}, \tag{1.10}$$

$$\frac{I_{d/2}(\hat{\kappa}_h)}{I_{d/2-1}(\hat{\kappa}_h)} = \frac{\|\mathbf{r}_h\|}{\sum_{i=1}^{n} p(h|\mathbf{x}_i,\Theta)}. \tag{1.11}$$

Observe that (1.10) and (1.11) are intuitive generalizations of (1.4) and (1.5) 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 [34, 11] is given by

$$p(h|\mathbf{x}_i,\Theta) = \frac{\alpha_h\, f_h(\mathbf{x}_i|\Theta)}{\sum_{l=1}^{k} \alpha_l\, f_l(\mathbf{x}_i|\Theta)}. \tag{1.12}$$

It can be shown [15] 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 1.6.

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

$$q(h|\mathbf{x}_i,\Theta) = \begin{cases} 1, & \text{if } h = \underset{h'}{\operatorname{argmax}}\ \ p(h'|\mathbf{x}_i,\Theta), \\ 0, & \text{otherwise.} \end{cases} \tag{1.13}$$

It can be shown [5] that the above hard-assignment rule actually maximizes a non-trivial 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 1.6.

1.5 Handling High-dimensional Text Datasets

Although the mixture model outlined in Section 1.4 appears to be straightforward, there is one critical issue that needs to be addressed before one can apply the model to real life text datasets: How to efficiently and accurately compute $\kappa_h, h = 1, \ldots, k$ from (1.11) for high-dimensional data? The problem of estimating κ_h is analyzed in Section 1.5.1 and experimentally studied in Section 1.5.2.

1.5.1 Approximating κ

Recall that due to the lack of an analytical solution, it is not possible to directly estimate the κ values (see (1.5) and (1.11)). 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 [30] provide approximations for estimating κ for a single component (1.5) for two limiting cases (Approximations (10.3.7) and (10.3.10) of [30, pp. 198]):

$$\hat{\kappa} \approx \frac{d-1}{2(1-\bar{r})} \qquad \text{valid for large } \bar{r}, \tag{1.14}$$

$$\hat{\kappa} \approx d\bar{r}\left(1 + \frac{d}{d+2}\bar{r}^2 + \frac{d^2(d+8)}{(d+2)^2(d+4)}\bar{r}^4\right) \qquad \text{valid for small } \bar{r}, \tag{1.15}$$

where $\bar{r}$ is given by (1.5).

These approximations assume that $\kappa \gg d$, which is typically not valid for high dimensional data (see the discussion in Section 1.8 for an intuition). Furthermore, the $\bar{r}$ values corresponding to the text datasets considered in this chapter 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 $A_d(\kappa)$ [52] given by

$$A_d(\kappa) = \frac{I_{d/2}(\kappa)}{I_{d/2-1}(\kappa)} = \cfrac{1}{\frac{d}{\kappa} + \cfrac{1}{\frac{d+2}{\kappa} + \cdots}}\,. \tag{1.16}$$

Letting $A_d(\kappa) = \bar{r}$, we can write (1.16) approximately as

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

which yields

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

We empirically found (see Section 1.5.2 below) 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

$$\hat{\kappa} = \frac{\bar{r}d - \bar{r}^3}{1-\bar{r}^2} \ . \tag{1.17}$$

Recently Tanabe et al. [50] used some inequalities regarding the Bessel function ratio $A_d(\kappa)$ [3] to bound the solution to $A_d(\kappa) = \bar{r}$ as

$$\frac{\bar{r}(d-2)}{1-\bar{r}^2} \leq \hat{\kappa} \leq \frac{\bar{r}d}{1-\bar{r}^2}.$$

Our solution (1.17) lies within these bounds, thus leading to a better theoretical justification in retrospect.

The approximation in (1.17) could perhaps be made even more accurate by adding other correction terms that are functions of $\bar{r}$ and d. However, we remark that if one wants a more accurate approximation, it is easier to use (1.17) 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 (1.17), and a very high accuracy for κ is not that critical. For other approximations of κ and some related issues, the reader is referred to [21, 5].

We now show some numerical results to assess the quality of our approximation in comparison to (1.14) and (1.15). First note that a particular value of $\bar{r}$ may correspond to many different combinations of κ and d values. Then, 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 1.5.2 below 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 set of points on this plane via Table 1.1 which shows the actual numerical values of κ that the three approximations (1.14), (1.15), and (1.17) yielded at these points. The $\bar{r}$ values shown in the table were computed using (1.5).

1.5.2 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

$(d, \bar{r}, \kappa)$	$\hat{\kappa}$ in (1.14)	$\hat{\kappa}$ in (1.15)	$\hat{\kappa}$ in (1.17)
$(10, 0.633668, 10)$	12.3	9.4	**10.2**
$(100, 0.46945, 60)$	93.3	59.4	**60.1**
$(500, 0.46859, 300)$	469.5	296.8	**300.1**
$(1000, 0.554386, 800)$	1120.9	776.8	**800.1**

Table 1.1: Approximations $\hat{\kappa}$ for a sampling of κ and d values.

approximation (1.17) attempts to solve the implicit non-linear equation

$$\frac{I_{d/2}(\kappa)}{I_{d/2-1}(\kappa)} = \bar{r}. \tag{1.18}$$

We note that for large values of $\bar{r}$ ($\bar{r}$ close to 1), approximation (1.14) is reasonable; for small values of $\bar{r}$ (usually for $\bar{r} < 0.2$) estimate (1.15) is quite good; whereas (1.17) yields good approximations for most values of $\bar{r}$.

Since a particular value of $\bar{r}$ may correspond to many different combinations of κ and d values, 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 1.1, 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 (1.18).

We begin by holding d fixed at 1000, and allow κ to vary from 10 to 5010. Figure 1.1 shows the values of computed $\hat{\kappa}$ (estimation of κ) using the three approximations. From this figure one can see that (1.14) overestimates the true κ, while (1.15) underestimates it. However, our approximation (1.17) 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 1.2 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 (1.15) catches up with the true value of κ after approximately $d \geq 2\kappa$ (because the associated $\bar{r}$ values become small), whereas (1.17) 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 1.3 shows how the three 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 (1.14), (1.15) and (1.17), respectively. Figure 1.3 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

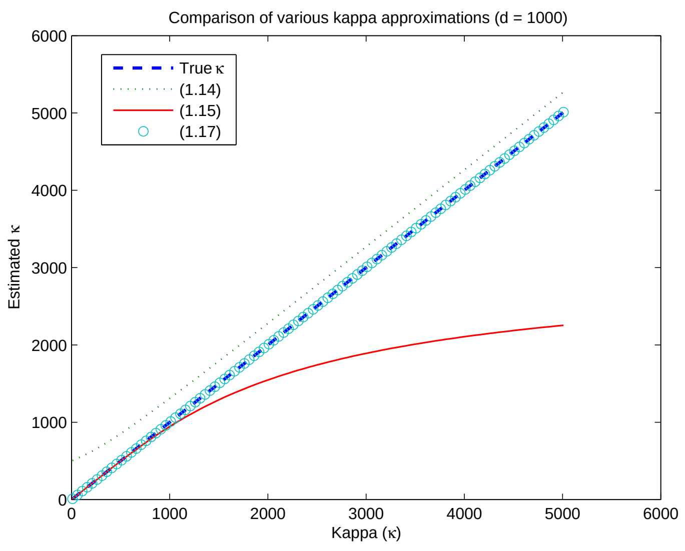


FIGURE 1.1: Comparison of true and approximated κ values, with $d = 1000$

on the plot), the approximation yielded by (1.15) has lower error. Thereafter, approximation (1.17) becomes better.

1.6 Algorithms

Mixture models based on vMF distributions 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 1.4 using EM. Hence, it assigns soft (or probabilistic) labels to each point that are given by the posterior probabilities of the components of the mixture conditioned on the point. On termination, the algorithm gives the parameters $\Theta = \{\alpha_h, \mu_h, \kappa_h\}_{h=1}^{k}$ of the k vMF distributions that model the dataset $\mathcal{X}$, as well as the *soft-clustering*, i.e., the posterior probabilities $p(h|\mathbf{x}_i, \Theta)$, 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 (1.13). After the hard assignments in every iteration,

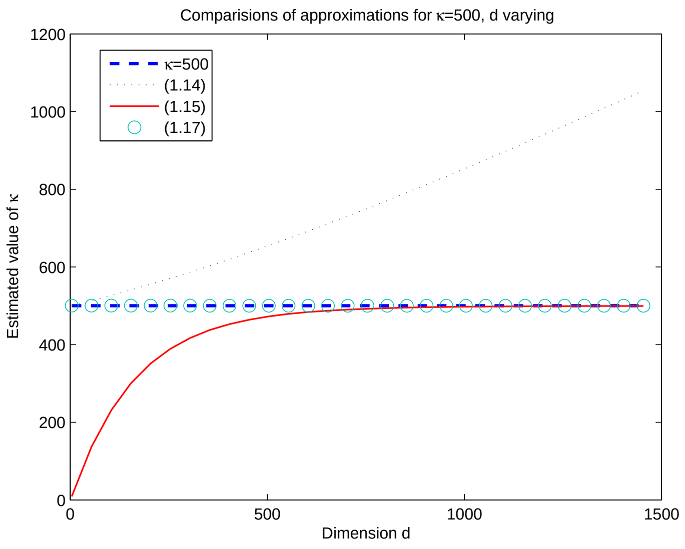


FIGURE 1.2: Comparison of approximations for varying d, $\kappa = 500$.

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 $\Theta = \{\alpha_h, \mu_h, \kappa_h\}_{h=1}^k$ of the k vMFs that model the input dataset $\mathcal{X}$.

Finally, we show that by 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. In a mixture of vMF model, assume that the priors of all the components are equal, i.e., $\alpha_h = 1/k, \forall h$, and that all the components have (equal) infinite concentration parameters, i.e., $\kappa_h = \kappa \to \infty, \forall h$. Under these assumptions the E-step in the `soft-moVMF` 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, i.e., a point $\mathbf{x}_i$ will be assigned to cluster $h^* = \text{argmax}_h \ \mathbf{x}_i^T \mu_h$, since

$$p(h^*|\mathbf{x}_i, \Theta) = \lim_{\kappa \to \infty} \frac{e^{\kappa \ \mathbf{x}_i^T \mu_{h^*}}}{\sum_{h=1}^k e^{\kappa \ \mathbf{x}_i^T \mu_h}} = 1,$$

and $p(h|\mathbf{x}_i, \Theta) \to 0$, 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,

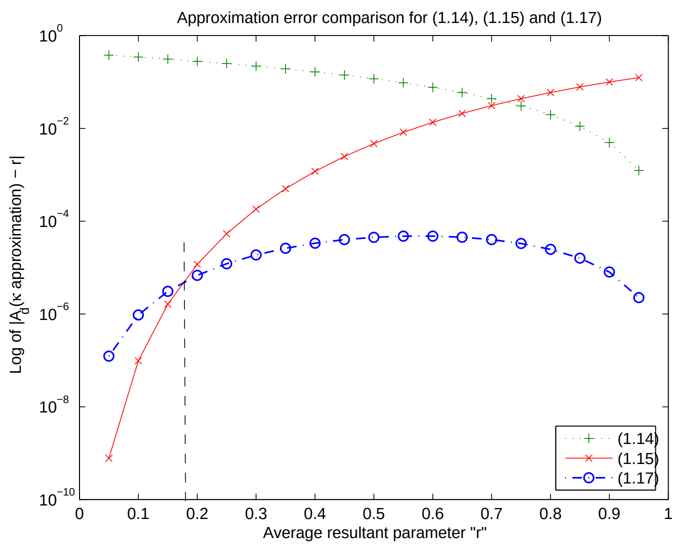


FIGURE 1.3: Comparison of approximations for varying $\bar{r}$ (with $d = 1000$)

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 $\mathbf{x}_i^T \mu_h$, and `hard-moVMF` reduces to `spkmeans`.

In addition to the above mentioned algorithms, we report experimental results on another algorithm `fskmeans` [6] 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 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 [14].

1.7 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 several datasets.

Algorithm 1 `soft-moVMF`

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

1. Spherical KMeans [20]—`spkmeans`.
2. Frequency Sensitive Spherical KMeans [6]—`fskmeans`.
3. moVMF based clustering using hard assignments—`hard-moVMF`.
4. moVMF based clustering using soft assignments—`soft-moVMF`.

It has already been established that `kmeans` using Euclidean distance performs much worse than `spkmeans` for text data [49], 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 datasets, have also not been included in the experiments. This exclusion is done as a recent empirical study over 15 text datasets showed that simple versions of vMF mixture models (with κ constant for all clusters) outperform the multinomial model except for only one dataset (Classic3), and the Bernoulli model was inferior for all datasets [56]. Further, for certain datasets, we compare clustering performance with latent Dirichlet allocation (LDA) [12] and exponential family approximation of Dirichlet compounded multinomial (EDCM) models [23].

Algorithm 2 `hard-moVMF`

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

1.7.1 Datasets

The datasets 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 datasets 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: Simulated, Classic3, Yahoo News, 20 Newsgroups, and Slashdot. For all the text document datasets, the toolkit MC [17] 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 the simulated datasets.

- **Simulated.** 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 [53] to generate a set of data points following a given vMF distribution. We describe herein, two synthetic datasets. The first dataset **small-mix** is 2-dimensional and is used to illustrate soft-clustering. The second dataset **big-mix** is a high-dimensional dataset

Algorithm 3 `spkmeans`

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

that could serve as a model for real world text datasets. 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:** 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.** This is a well known collection of documents. It is an easy dataset to cluster since it contains documents from three well-separated sources. Moreover, the intrinsic clusters are largely balanced.

1. **Classic3** is a corpus containining 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** 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 dataset. The dimensionality of the data was 5471.[2]

3. **Classic400** is a subset of Classic3 that has 400 documents. This dataset has 100 randomly chosen documents from the MEDLINE and CISI categories and 200 randomly chosen documents from the CRANFIELD category. This dataset is specifically designed to create unbalanced clusters in an otherwise easily separable and balanced dataset. The dimensionality of the data was 6205.

- **Yahoo News (K-series).** This compilation has 2340 Yahoo news articles from 20 different categories. The underlying clusters in this dataset 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.

- **20 Newsgroup.** The 20 Newsgroup dataset is a widely used compilation of documents [28]. We tested our algorithms on not only the original dataset, but on a variety of subsets with differing characteristics to explore and understand the behavior of our algorithms.

  1. **News20** is a standard dataset that comprises 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 dataset—high-dimensionality, sparsity and significantly overlapping clusters.
  2. **Small-news20** is formed by selecting 2000 messages from the original News20 dataset. We randomly selected 100 messages from each category in the original dataset. Hence this dataset 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,mideast,misc}.

[2]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.

5. **Different-100/1000** is a collection of 100/1000 messages from 3 very different newsgroups: alt.atheism, rec.sport.baseball, sci.space.

- **Slash-dot.** We harvested news articles from the Slashdot website and created 2 datasets. For each category in these datasets, we collected 1000 articles primarily tagged with the category label, and then removed articles that were posted to multiple categories.

  1. **Slash-7** contains 6714 news articles posted to 7 Slashdot categories: Business, Education, Entertainment, Games, Music, Science and Internet.
  2. **Slash-6** contains 5182 articles posted to the 6 categories: Biotech, Microsoft, Privacy, Google, Security, Space.

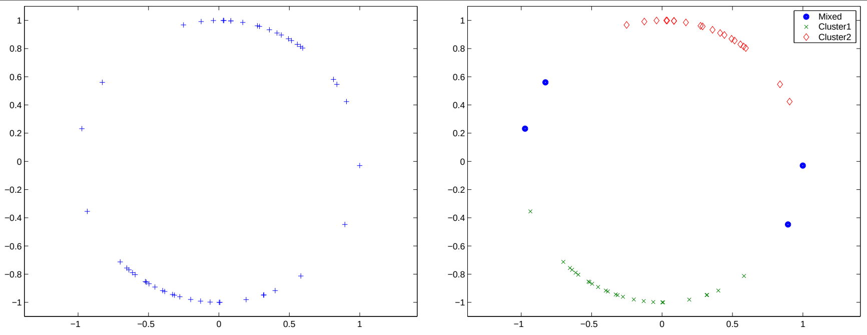
1.7.2 Methodology

Performance of the algorithms on all the datasets has been analyzed using *mutual information* (MI) between the cluster and class labels. MI quantifies the amount of statistical similarity between the cluster and class labels [16]. 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 dataset under consideration. To facilitate computing MI, 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). Note that variants of MI have been used to evaluate clustering algorithms by several researchers. The authors of [32] 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 [22].

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 multivariate Gaussian in a mixture of Gaussians remains non-singular.

1.7.3 Simulated Datasets

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 dataset shown in Figure 1.4(a).



(a)
The small-mix dataset.

(b)
A clustering of small-mix.

FIGURE 1.4: Small-mix dataset and its clustering by `soft-moVMF`.

The clustering produced by our soft cluster assignment algorithm is shown in Figure 1.4(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 dataset 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 dataset with well-separated clusters. Further testimony to `soft-moVMF`'s performance is served by Table 1.2, which shows the discrepancy between true and estimated parameters for the small-mix collection.

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

Before moving onto real datasets let us briefly look at the behavior of the algorithms on the larger dataset big-mix. On calculating MI as described

Table 1.2: True and estimated parameters for small-mix using `soft-moVMF`.

Cluster	μ	$\hat{\mu}$	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

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

Table 1.3: Performance of `soft-moVMF` on big-mix dataset.

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

availability of a sufficient number of data points and similar sized clusters. For reference Table 1.3 offers numerical evidence about the performance of `soft-moVMF` on the big-mix dataset.

1.7.4 Classic3 Family of Datasets

Table 1.4 shows typical confusion matrices obtained for the full Classic3 dataset. 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 1.4: 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
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 1.5: Comparative confusion matrices for 3 clusters of Classic300.

`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 1.5 shows the confusion matrices obtained for the Classic300 dataset. 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.

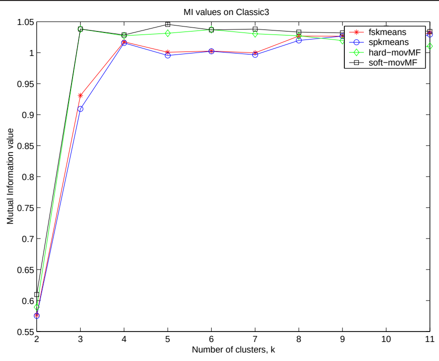
Table 1.6: Comparative confusion matrices for 3 clusters of Classic400.

`fskmeans`			`spkmeans`			`hard-moVMF`			`soft-moVMF`		
med	cisi	cran	med	cisi	cran	med	cisi	cran	med	cisi	cran
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**

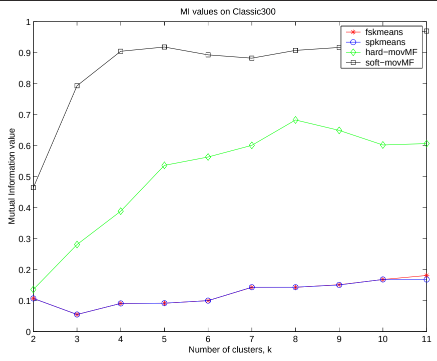
It seems that the low number of documents does not pose a problem for `soft-moVMF` and it ends up getting an almost perfect clustering for this dataset. Thus in this case, despite the low number of points per cluster, the superior modeling power of our moVMF based algorithms prevents them from getting trapped in inferior local-minima as compared to the other algorithms—resulting in a better clustering.

The confusion matrices obtained for the Classic400 dataset are displayed in Table 1.6. The behavior of the algorithms for this dataset 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.

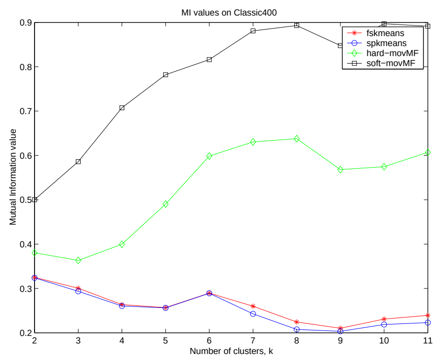
An insight into the working of the algorithms is provided by considering their clustering performance when they are requested to produce greater than



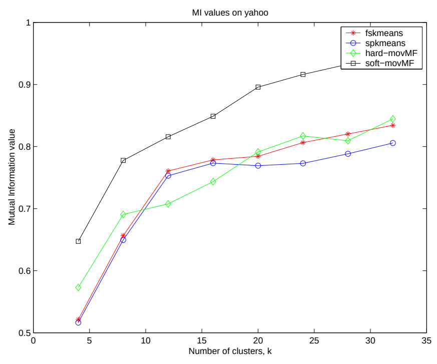
(a) MI values for Classic3.



(b) MI values for Classic300.



(c) MI values for Classic400.



(d) MI values for Yahoo News.

FIGURE 1.5: Comparison of the algorithms for the Classic3 datasets and the Yahoo News dataset.

the "natural" number of clusters. In Table 1.7 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 1.6, suggest a clustering method in which one could generate a slightly higher number of clusters than required, and then agglomerate them appropriately.

Table 1.7: Comparative confusion matrices for 5 clusters of Classic3.

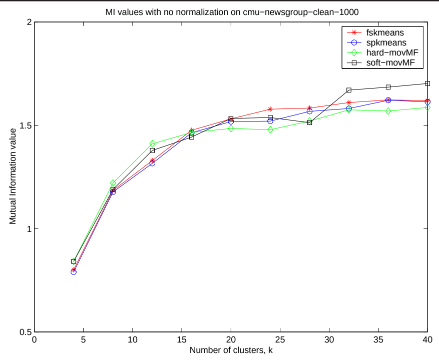
`fskmeans`			`spkmeans`			`hard-moVMF`			`soft-moVMF`		
med	cisi	cran	med	cisi	cran	med	cisi	cran	med	cisi	cran
2	4	**312**	2	4	**323**	3	5	**292**	0	1	**1107**
8	**520**	10	8	**512**	9	**511**	1	0	5	**1455**	14
5	**936**	6	5	**944**	6	**514**	1	0	**526**	2	1
1018	0	1	**1018**	0	1	0	2	**1093**	**501**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

The MI plots for the various Classic3 datasets are given in Figures 1.5(a)-(c). For the full Classic3 dataset (Figure 1.5(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 1.5(b)) and Classic400 (Figure 1.5(c)), `soft-moVMF` seems to significantly outperform the other algorithms. In fact, for these two datasets, `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.

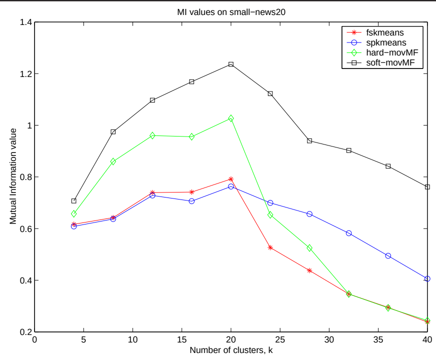
1.7.5 Yahoo News Dataset

The Yahoo News dataset is a relatively difficult dataset 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.

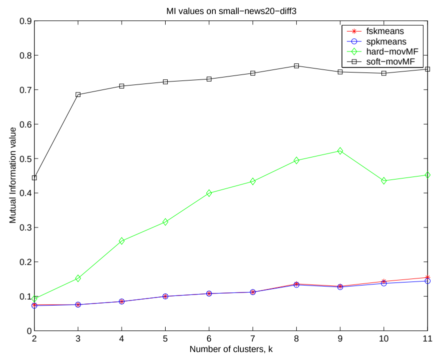
Results for the different algorithms can be seen in Figure 1.5(d). Over the entire range, `soft-moVMF` consistently performs better than the other algorithms. Even at the correct number of clusters $k = 20$, it performs significantly better than the other algorithms.



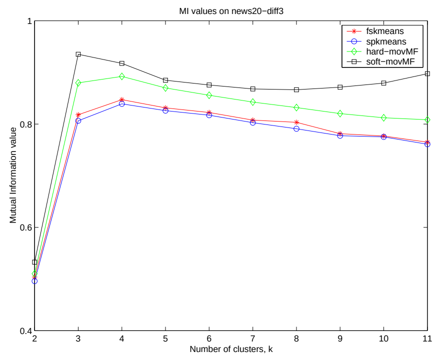
(a) MI values for News20.



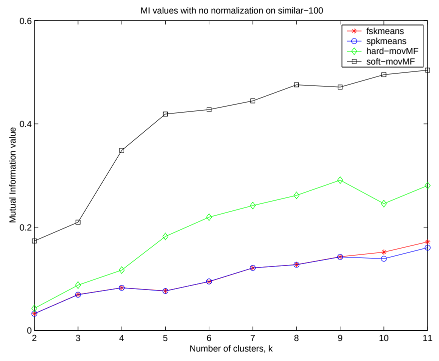
(b) MI values for Small-news20.



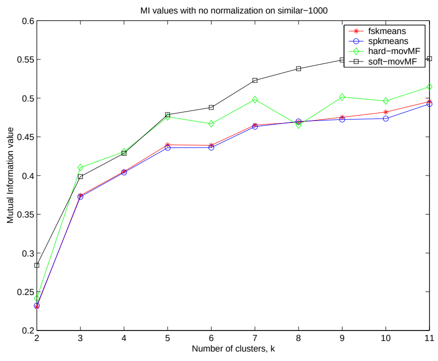
(c) MI values for Different-100.



(d) MI values for Different-1000.

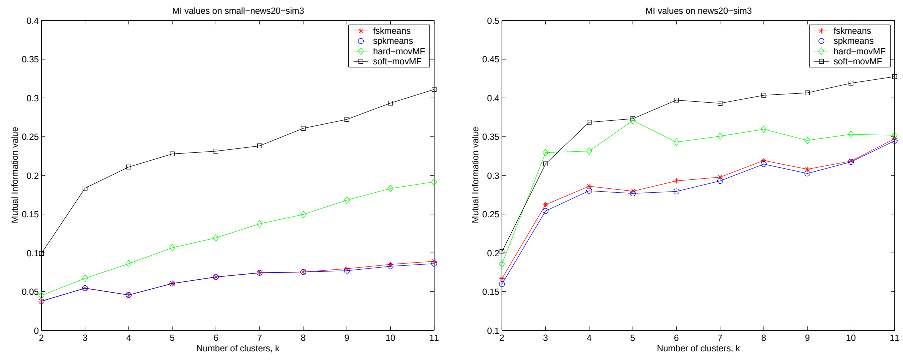


(e) MI values for Similar-100.



(f) MI values for Similar-1000.

FIGURE 1.6: Comparison of the algorithms for the 20 Newsgroup and some subsets.



(a) MI values for Same-100.   (b) MI values for Same-1000.

FIGURE 1.7: Comparison of the algorithms for more subsets of 20 Newsgroup data.

1.7.6 20 Newsgroup Family of Datasets

Now we discuss clustering performance of the four algorithms on the 20 Newsgroup datasets. Figure 1.6(a) shows the MI plots for the full News20 dataset. 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 1.6(b) shows MI plots for the Small-News20 dataset 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 datasets 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 1.6(d), 1.6(f) and 1.7(b). We note that the other three algorithms perform quite similarly over the entire range of clusters. We also observe that for an easy dataset like Different-1000, the MI values peak at the true number of clusters, whereas for a more difficult dataset 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.

1.7.7 Slashdot Datasets

The Slashdot dataset was created to test the performance of the moVMF model on a typical web application. To gain a better understanding of the relative performance of the model compared to other state-of-the-art models for text clustering and topic modeling, moVMF was compared with latent Dirichlet allocation (LDA) [12] and the exponential family approximation of the Dirichlet compounded multinomial (EDCM) model [23]. Table 1.8 shows the comparative performance in terms of cluster quality measured by normalized mutual information (NMI), and in terms of running time. Overall, moVMF gives significantly better clustering results, while the running time is an order of magnitude less compared to the other algorithms. Similar results on other benchmark datasets have been reported by [4].

Table 1.8: Performance comparison of algorithms averaged over 5 runs.

	NMI			Run Time (sec)		
Dataset	moVMF	EDCM	LDA	vMF	EDCM	LDA
slash-7	**0.39**	0.22	0.31	15	40	47
slash-6	**0.65**	0.36	0.46	6	26	36

Table 1.9 shows the qualitative performance of moVMF model on the Slash-7 dataset in terms of the top keywords associated with five of the clusters. The "topics" associated with each cluster is of comparable quality to that generated by Bayesian topic models such as LDA [4].

Table 1.9: Five of the topics obtained by running batch vMF on slash-7.

music	web	scientists	internet	games
apple	google	nasa	broadband	gaming
itunes	search	space	domain	game
riaa	yahoo	researchers	net	nintendo
ipod	site	science	network	sony
wikipedia	online	years	verisign	xbox
digital	sites	earth	bittorrent	gamers
napster	ebay	found	icann	wii
file	amazon	brain	service	console
drm	engine	university	access	video

1.8 Discussion

The mixture of vMF distributions gives a parametric model-based generalization of the widely used cosine similarity measure. As discussed in Section 1.6, 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 [2]. More precisely, consider a dataset 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 [25] corresponding to the vMF distribution is given by

$$\begin{aligned} K(\mathbf{x}_i, \mathbf{x}_j) &= (\nabla_\mu \ln f(\mathbf{x}_i|\mu))^T (\nabla_\mu \ln f(\mathbf{x}_j|\mu)) \quad \text{(see (1.1))} \\ &= (\nabla_\mu(\mu^T \mathbf{x}_i))^T (\nabla_\mu(\mu^T \mathbf{x}_j)) = \mathbf{x}_i^T \mathbf{x}_j, \end{aligned}$$

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, a couple of issues appear intriguing: (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 1.8. The behavior is similar for all the other datasets 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. Thus, 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 updates, 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

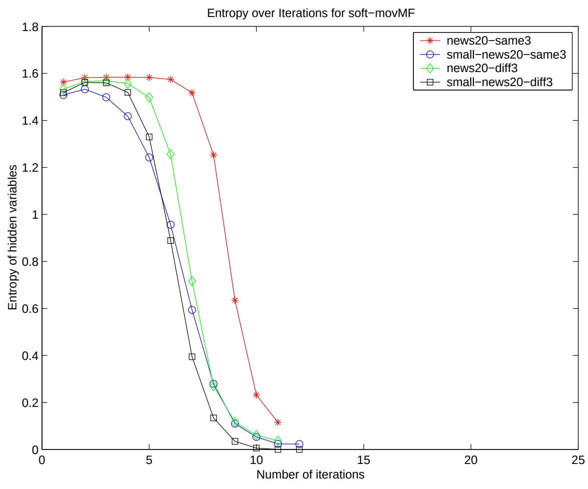


FIGURE 1.8: Variation of Entropy of hidden variables with number of Iterations (`soft-movMF`).

model, suffers because of hard-assignments from the very beginning. The `fskmeans` and `spkmeans` do not do well for difficult datasets 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 [31]. A new objective criterion for evaluation and model-selection for clustering algorithms was proposed in [8]: how well does the clustering algorithm perform as a prediction algorithm. The prediction accuracy of the clustering is measured by the PAC-MDL bound [13, 8] 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 [8]. Interestingly, the movMF-based algorithms almost always obtained the 'right number of clusters'—in this case, the underlying labels in the dataset 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.

1.9 Conclusions and Future Work

From the experimental results, it seems that high-dimensional text 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 [34]. 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 [18], 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 (already discussed in the previous section), the Fisher-Bingham distribution, the Pearson type VII distributions [42, 30], 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 [36]. 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.

Acknowledgements

The authors would like to thank Sugato Basu and Jiye Yu for experiments with the Slashdot datasets. This research was supported in part by the Digital Technology Center Data Mining Consortium (DDMC) at the University of Minnesota, Twin Cities.

References

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

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

[3] D. E. Amos. Computation of modified Bessel functions and their ratios. *Mathematics of Computation*, 28(125):235–251, 1974.

[4] A. Banerjee and S. Basu. Topic models over text streams: A study of batch and online unsupervised learning. In *Proceedings of the 7th SIAM International Conference on Data Mining*, 2007.

[5] A. Banerjee, I. Dhillon, J. Ghosh, and S. Sra. Clustering on the unit hypersphere using von Mises-Fisher distributions. *Journal of Machine Learning Research*, 6:1345–1382, 2005.

[6] 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.

[7] 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.

[8] 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.

[9] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005.

[10] A. Bijral, M. Breitenbach, and G. Z. Grudic. Mixture of Watson Distributions: A Generative Model for Hyperspherical Embeddings. In *AISTATS*, 2007.

[11] 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.

[12] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

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

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

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

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

[17] 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.

[18] 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*, 2002.

[19] I. S. Dhillon, E. M. Marcotte, and U. Roshan. Diametrical clustering for identifying anti-correlated gene clusters. *Bioinformatics*, 19(13):1612–1619, 2003.

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

[21] 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.

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

[23] C. Elkan. Clustering documents with an exponential-family approximation of the Dirichlet compund multinomial distribution. In *Proceedings of the 23rd International Conference on Machine Learning*, 2006.

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

[25] 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.

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

[27] 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.

[28] K Lang. News Weeder: Learning to filter netnews. In *Proceedings 12th International Conference on Machine Learning*, pages 331–339, San Francisco, 1995.

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

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

[31] 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.

[32] M. Meilă. Comparing clusterings by the variation of information. In *Proceedings of the 16th Annual Conference on Learning Theory*, 2003.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

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

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

[39] 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.

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

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

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

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

[44] 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.

[45] S. Sra. *Matrix Nearness Problems in Data Mining.* PhD thesis, The University of Texas at Austin, August 2007.

[46] A. N. Srivastava and R. Akella. Enabling the discovery of recurring anomalies in aerospace system problem reports using high-dimensional clustering techniques. In *Proceedings of the IEEE Aerospace Conference*, 2006.

[47] A. N. Srivastava and B. Zane-Ulman. Discovering hidden anomalies in text reports regarding complex space systems. In *IEEE Aerospace Conference*, 2005.

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

[49] 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.

[50] A. Tanabe, K. Fukumizu, S. Oba, T. Takenouchi, and S. Ishii. Parameter estimation for von Mises-Fisher distributions. *Computational Statistics*, 22(1):145–157, 2007.

[51] 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.

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

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

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

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

[56] 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 2003.