Journal of Statistical Software

July 2014, Volume 58, Issue 10. http://www.jstatsoft.org/

movMF: An R Package for Fitting Mixtures of von Mises-Fisher Distributions

Kurt Hornik
WU Wirtschaftsuniversität Wien

Bettina Grün
Johannes Kepler Universität Linz

Abstract

Finite mixtures of von Mises-Fisher distributions allow to apply model-based clustering methods to data which is of standardized length, i.e., all data points lie on the unit sphere. The R package **movMF** contains functionality to draw samples from finite mixtures of von Mises-Fisher distributions and to fit these models using the expectation-maximization algorithm for maximum likelihood estimation. Special features are the possibility to use sparse matrix representations for the input data, different variants of the expectation-maximization algorithm, different methods for determining the concentration parameters in the M-step and to impose constraints on the concentration parameters over the components.

In this paper we describe the main fitting function of the package and illustrate its application. In addition we compare the clustering performance of finite mixtures of von Mises-Fisher distributions to spherical k-means. We also discuss the resolution of several numerical issues which occur for estimating the concentration parameters and for determining the normalizing constant of the von Mises-Fisher distribution.

Keywords: EM algorithm, finite mixture, hypergeometric function ${}_0F_1$, modified Bessel function ratio, R, von Mises-Fisher distribution.

1. Introduction

Finite mixture models allow to cluster observations by assuming that for each component a suitable parametric distribution can be specified and that the mixture distribution is derived by convex combination of the component distributions. McLachlan and Peel (2000) and Frühwirth-Schnatter (2006) present overviews on the estimation of these models in a maximum likelihood as well as a Bayesian setting together with different applications of finite mixture models.

If the support of the data is given by the unit sphere, a natural choice for the component

distributions is the von Mises-Fisher (vMF) distribution. The special case where data is in $\mathbb{R}^2$, i.e., the observations lie on the unit circle, is referred to as von Mises distribution. The vMF distribution has concentric contour lines similar to the multivariate normal distribution with the variance-covariance matrix being a multiple of the identity matrix. In fact Mardia and Jupp (1999, p. 173) point out that if X follows a multivariate normal distribution with mean parameter μ which has length one and variance-covariance matrix $\kappa^{-1}I$, then the conditional distribution of X given that it has length one is a vMF distribution with mean direction parameter μ and concentration parameter κ.

Finite mixtures of vMF distributions were introduced in Banerjee, Dhillon, Ghosh, and Sra (2005) to cluster data on the unit sphere. They propose to use the expectation-maximization (EM) algorithm for maximum likelihood (ML) estimation and present two variants which they refer to as hard and soft clustering. The areas of application of the presented examples include text mining where abstracts of scientific journals, news articles and newsgroup messages are categorized, and bioinformatics using a yeast gene expression data set. In these applications, available data is typically high-dimensional. The estimation methods employed need to take this into account. For finite mixtures of vMF distributions this is particularly relevant for the determination of the concentration parameters.

Tang, Chu, and Huang (2009) use finite mixtures of vMF distributions for speaker clustering. The data is pre-processed by fitting a Gaussian mixture model (GMM) to the utterances and stacking the mean vectors of the mixture components to form the GMM mean supervector which is then used as input for the mixture model aiming at clustering the speakers. They compare the performance of finite mixtures of Gaussian distributions with those of vMF distributions and conclude that for this application the latter outperform finite mixtures of multivariate normal distributions. Further possible areas of application for finite mixtures of vMF distributions are segmentation in high angular resolution diffusion imaging (McGraw, Vemuri, Yezierski, and Mareci 2006) and clustering treatment beam directions in external radiation therapy (Bangert, Hennig, and Oelfke 2010).

This paper is structured as follows. Section 2 introduces the vMF distribution and shows how to draw samples from this distribution and how to determine the ML estimates of its parameters. The extension to finite mixture models is covered in Section 3 including again the methods for drawing samples and determining ML estimates. The R (R Core Team 2014) package **movMF** (Hornik and Grün 2014) is presented in Section 4 by describing the main fitting function `movMF()` in detail. The package is available from the Comprehensive R Archive Network at `http://CRAN.R-project.org/package=movMF`. Numerical issues when evaluating the density or determining the ML estimates are discussed in Section 5. Section 6 compares the performance of mixtures of vMF distributions with spherical k-means clustering using a standard set of data sets from text mining. An illustrative application of the package using the abstracts from the talks at "useR! 2008", the 3rd international R user conference, in Dortmund, Germany, is given in Section 7. The paper concludes with a summary and an outlook on possible extensions.

2. The vMF distribution

A random unit length vector in $\mathbb{R}^d$ has a *von Mises-Fisher* (or Langevin) distribution with parameter $\theta \in \mathbb{R}^d$ if its density with respect to the uniform distribution on the unit sphere

$\mathbb{S}^{d-1} = \{x \in \mathbb{R}^d : \|x\| = 1\}$ is given by

$$f(x|\theta) = e^{\theta^\top x}/{}_0F_1(;d/2;\|\theta\|^2/4),$$

where

$${}_0F_1(;\nu;z) = \sum_{n=0}^{\infty} \frac{\Gamma(\nu)}{\Gamma(\nu+n)} \frac{z^n}{n!}$$

is the confluent hypergeometric limit function and related to the modified Bessel function of the first kind I_ν via

$${}_0F_1(;\nu+1;z^2/4) = \frac{I_\nu(z)\Gamma(\nu+1)}{(z/2)^\nu} \tag{1}$$

(e.g., Mardia and Jupp 1999, p. 168).

The vMF distribution is commonly parametrized as $\theta = \kappa\mu$, where $\kappa = \|\theta\|$ and $\mu \in \mathbb{S}^{d-1}$ are the *concentration* and *mean direction* parameters, respectively (if $\theta \neq 0$, μ is uniquely determined as $\theta/\|\theta\|$).

In what follows, it will be convenient to write

$$C_d(\kappa) = 1/{}_0F_1(;d/2;\kappa^2/4)$$

so that the density is given by

$$f(x|\theta) = C_d(\|\theta\|)e^{\theta^\top x}.$$

2.1. Simulating vMF distributions

The following algorithm provides a rejecting sampling scheme for drawing a sample from the vMF distribution with modal direction $(0,\ldots,0,1)$ and concentration parameter $\kappa \geq 0$ (see Algorithm VM* in Wood 1994). The extension for simulating from the matrix Bingham vMF distribution is described in Hoff (2009) and also available in R through package **rstiefel** (Hoff 2012).

Step 1. Calculate b using

$$b = \frac{d-1}{2\kappa + \sqrt{4\kappa^2 + (d-1)^2}}.$$

Note that this calculation of b is algebraically equivalent to the one proposed in Wood (1994), but numerically more stable.

Put $x_0 = (1-b)/(1+b)$ and $c = \kappa x_0 + (d-1)\log(1-x_0^2)$.

Step 2. Generate $Z \sim \text{Beta}((d-1)/2,(d-1)/2)$ and $U \sim \text{Unif}([0,1])$ and calculate

$$W = \frac{1-(1+b)Z}{1-(1-b)Z}.$$

Step 3. If

$$\kappa W + (d-1)\log(1-x_0W) - c < \log(U),$$

go to Step 2.

Step 4. Generate a uniform $(d-1)$-dimensional unit vector V and return

$$X = (\sqrt{1-W^2}V^\top, W)^\top.$$

The uniform $(d-1)$-dimensional unit vector V can be generated by simulating independent standard normal random variables and normalizing them (see for example Ulrich 1984). To get samples from a vMF distribution with arbitrary mean direction parameter μ, X is multiplied from the left with a matrix where the first $(d-1)$ columns consist of unitary basis vectors of the subspace orthogonal to μ and the last column is equal to μ.

2.2. Estimating the parameters of the vMF distribution

Using the common parametrization by κ and μ, the log-likelihood of a sample $x_1, \ldots, x_n$ from the vMF distribution is given by

$$n \log(C_d(\kappa)) + \kappa \mu^\top r,$$

where $r = \sum_{i=1}^n x_i$ is the resultant vector (sum) of the x_i. The maximum likelihood estimates (MLEs) are obtained by solving the likelihood equations

$$\hat{\mu} = r/\|r\|, \qquad -\frac{C_d'(\hat{\kappa})}{C_d(\hat{\kappa})} = \|r\|/n.$$

Writing $A_d(\kappa) = -C_d'(\kappa)/C_d(\kappa)$ for the logarithmic derivative of $1/C_d(\kappa)$ and $\rho = \|r\|/n$ for the average resultant length, the equation for the MLE of κ becomes

$$A_d(\hat{\kappa}) = \rho.$$

Using recursions for the modified Bessel function (e.g., Watson 1995, p. 71), one can establish that

$$A_d(\kappa) = -\frac{C_d'(\kappa)}{C_d(\kappa)} = \frac{I_{d/2}(\kappa)}{I_{d/2-1}(\kappa)}.$$

It can be shown (see for example Schou 1978) that A_d is a strictly increasing function which maps the interval $[0, \infty)$ onto the interval $[0, 1)$ and satisfies the Riccati equation $A_d'(\kappa) = 1 - A_d(\kappa)^2 - \frac{d-1}{\kappa} A_d(\kappa)$. As A_d and hence its derivatives can efficiently be computed using continued fractions (see Section 5 for details), the equation $A_d(\hat{\kappa}) = \rho$ can efficiently be solved by standard iterative techniques provided that good starting approximations are employed.

Dhillon and Sra (2003) and subsequently Banerjee *et al.* (2005) suggest the approximation

$$\hat{\kappa} \approx \frac{\rho(d-\rho^2)}{1-\rho^2} \tag{2}$$

obtained by truncating the Gauss continued fraction representation of A_d and adding a correction term "determined empirically". The former reference also suggests using this approximation as the starting point of a Newton-Raphson iteration, using the above expression for A_d'.

Tanabe, Fukumizu, Oba, Takenouchi, and Ishii (2007) show that

$$\hat{\kappa} = \frac{\rho(d-c)}{1-\rho^2}$$

for some suitable $0 \leq c \leq 2$. The approximation in Equation 2 corresponds to $c \approx \rho^2$. They also suggest to determine $\hat{\kappa}$ via the fixed point iteration

$$\kappa_{t+1} = \kappa_t \rho / A_d(\kappa_t)$$

with a starting value in the solution range, e.g., using $c = 1$ or $c = \rho^2$.

Sra (2012) introduces a "truncated Newton approximation" based on performing exactly two Newton iterations

$$\kappa_{t+1} = \kappa_t - \frac{A_d(\kappa_t) - \rho}{A_d'(\kappa_t)}$$

with κ_0 determined via the $c = \rho^2$ approximation.

Song, Liu, and Wang (2012) suggest to use a "truncated Halley approximation" based on performing exactly two Halley iterations

$$\kappa_{t+1} = \kappa_t - \frac{2(A_d(\kappa_t) - \rho)A_d'(\kappa_t)}{2A_d'(\kappa_t)^2 - (A_d(\kappa_t) - \rho)A_d''(\kappa_t)}$$

with κ_0 determined via the $c = \rho^2$ approximation and using that the second derivative can be given as a function of $A_d(\kappa_t)$, i.e.,

$$A_d''(\kappa_t) = 2A_d(\kappa_t)^3 + \frac{3(d-1)}{\kappa} A_d(\kappa_t)^2 + \frac{d^2 - d - 2\kappa^2}{\kappa^2} A_d(\kappa_t) - \frac{d-1}{\kappa}.$$

The results in Hornik and Grün (2014) yield the substantially improved bounds

$$\max\left(F_{d/2-1,d/2+1}(\rho), F_{(d-1)/2,\sqrt{d^2-1}/4}(\rho)\right) \leq \hat{\kappa} \leq F_{(d-1)/2,(d+1)/2}(\rho), \tag{3}$$

valid for $0 \leq \rho < 1$, where

$$F_{\alpha,\beta}(\rho) = \frac{\rho}{1-\rho^2}\left(\alpha + \sqrt{\rho^2\alpha^2 + (1-\rho^2)\beta^2}\right).$$

The difference between the upper and lower bound is at most $3\rho/2$ for all $0 \leq \rho < 1$, and the difference between the lower bound and $\hat{\kappa}$ tends to 0 as $\rho \to 1-$.

Convex combinations of the lower and the upper bounds can be employed as starting values for the above iteration schemes (which require a single starting point). In addition, as these bounds actually give an interval known to contain the unique root $\hat{\kappa}$ of the function $\kappa \mapsto A_d(\kappa) - \rho$, one can use them as starting values for root finding methods which iteratively refine intervals containing the solution, such as simple bisection (as provided by `uniroot()` in R), hybrid algorithms combining derivative-based (Newton or Halley) and bisection steps (e.g. Press, Teukolsky, Vetterling, and Flannery 2002, p. 366), or the Newton-Fourier method (e.g. Atkinson 1989, pp. 62–64). One can show that A_d is concave (e.g., using Theorem 11 in Hornik and Grün 2013, which establishes that $A_d = R_{d/2-1}$ is the pointwise minimum of concave functions, and hence concave): hence, employing the above bounds and a variant of the Newton-Fourier method for strictly increasing concave functions yields a quadratically convergent iterative scheme for determining $\hat{\kappa}$.

2.3. Illustrative example: Household expenses

To illustrate the use of the vMF distribution to model data on the sphere we use the `household` data set from package **HSAUR2** (Everitt and Hothorn 2014). The data are part of a data set collected from a survey on household expenditures and give the expenses of 20 single men and 20 single women on four commodity groups. In the following we will focus only on three of those commodity groups (housing, food and service) to have 3-dimensional data which is easier to visualize. The data points are projected onto the sphere by normalizing them to have length one. Thus, in the following analysis we are interested in finding groups of households which lie in a similar direction, i.e., the angle between the observations is small and we are not interested in differences in their total absolute expenses.

The data points on the sphere are visualized in Figure 1 on the top left. Using the gender information, vMF distributions are fitted to the male and the female observations separately. The fitted distributions are visualized together with the mean direction indicated by a cross and with confidence circles of probability 50% (full lines) and 95% (dashed lines). Clearly the females have a smaller dispersion as indicated by the estimated κ which is equal to 96.4, as compared to the κ of the males which is given by 20.3.

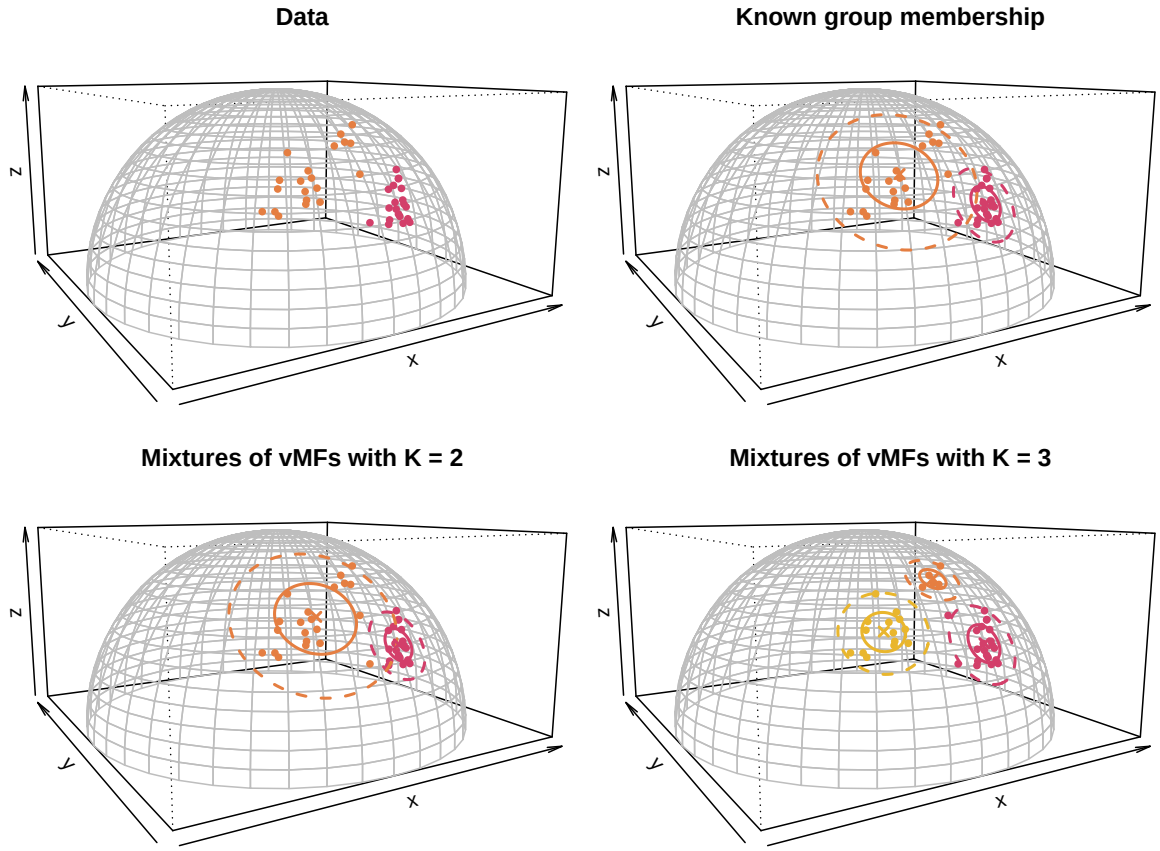

Figure 1: Household expenses data with gender indicated by color after projection to the sphere at the top left, estimated vMF distributions with confidence circles for each gender group separately at the top right and the estimated mixtures of vMF distributions with $K = 2$ and $K = 3$ with confidence circles at the bottom.

3. Finite mixtures of vMF distributions

The mixture model with K components is given by

$$h(x|\Theta) = \sum_{k=1}^{K} \pi_k f(x|\theta_k),$$

where $h(\cdot|\cdot)$ is the mixture density, Θ the vector with all π and θ parameters, and $f(y|\theta_k)$ the density of the vMF distribution with parameter θ_k. Furthermore, the component weights π_k are positive for all k and sum to one.

3.1. Simulating mixtures of vMF distributions

This is straightforwardly achieved by first sampling class ids $z \in \{1, \ldots, K\}$ with the mixture class probabilities $\pi_1, \ldots, \pi_K$, and then sampling the data from the respective vMF distributions with parameter θ_k.

3.2. Estimating the parameters of mixtures of vMF distributions

EM algorithms for ML estimation of the parameters of mixtures of vMF distributions are given in Dhillon and Sra (2003) and Banerjee *et al.* (2005). The EM algorithm exploits the fact that the complete-data log-likelihood where the component memberships of the observations are known is easier to maximize than the observed-data log-likelihood.

The EM algorithm for fitting mixtures of vMF distributions consists of the following steps:

1. Initialization: Either of the following two:

   (a) Assign values to π_k and θ_k for $k = 1, \ldots, K$, where $\pi_k > 0$ and $\sum_{k=1}^{K} \pi_k = 1$ and $\theta_k \neq \theta_l$ for all $k \neq l$ and $k, l = 1, \ldots, K$.

   Start the EM algorithm with an E-step.

   (b) Assign (probabilities of) component memberships to each of the n observations. E.g., the output from spherical k-means (Hornik, Feinerer, Kober, and Buchta 2012) can be used.

   Start the EM algorithm with an M-step.

2. Repeat the following steps until the maximum number of iterations is reached or the convergence criterion is met.

   E-step: Because the complete-data log-likelihood is linear in the missing data which correspond to the component memberships, the E-step only consists of calculating the a-posteriori probabilities, the probabilities of belonging to a component conditional on the observed values, using

   $$p(k|x_i, \Theta) \propto \pi_k f(x_i|\theta_k).$$

M-step: Maximize the expected complete-data log-likelihood by determining separately for each k:

$$\hat{\pi}_k = \frac{1}{n}\sum_{i=1}^{n} p(k|x_i, \Theta),$$

$$\hat{\mu}_k = \frac{\sum_{i=1}^{n} p(k|x_i, \Theta) x_i}{\| \sum_{i=1}^{n} p(k|x_i, \Theta) x_i \|}, \qquad -\frac{C_d'(\hat{\kappa}_k)}{C_d(\hat{\kappa}_k)} = \frac{\| \sum_{i=1}^{n} p(k|x_i, \Theta) x_i \|}{\sum_{i=1}^{n} p(k|x_i, \Theta)}.$$

$\hat{\kappa}_k$ can be determined via the approximation of Equation 2, or one of the improved methods discussed in Section 2.2.

Convergence check: Assess convergence by checking (either or both)

(a) if the relative absolute change in the log-likelihood values is smaller than a threshold ϵ_1;

(b) if the relative absolute change in parameters is smaller than a threshold ϵ_2.

If converged, stop the algorithm.

This corresponds to the soft-movMF algorithm on page 1357 in Banerjee *et al.* (2005). In addition they propose the hard-movMF algorithm on page 1358. The algorithm above can be modified to the hard-movMF algorithm by adding a hardening step between E- and M-step:

H-step: Replace the a-posteriori probabilities by assigning each observation with probability 1 to one of the components where its a-posteriori probability is maximum. Assuming the maximum is unique, this corresponds to

$$p(k|x_i, \Theta) = \begin{cases} 1, & \text{if } k = \operatorname{argmax}_h p(h|x_i, \Theta), \\ 0, & \text{otherwise.} \end{cases}$$

If the maximum is not unique, assignment is randomly with equal probability to one of the $k \in \operatorname{argmax}_h p(h|x_i, \Theta)$, i.e., ties are broken at random.

This algorithm is also referred to as classification EM algorithm in the literature (Celeux and Govaert 1992). A further variant of the EM algorithm also considered for example in Celeux and Govaert (1992) would be the stochastic EM where instead of an hardening step a stochastic step is added between E- and M-step:

S-step: Assign at random each observation to one component with probability equal to its a-posteriori probability.

The algorithm above determines the parameter estimates if the concentration parameters are allowed to vary freely over components. An alternative model specification could be to impose the restriction that the concentration parameters are the same for all components. This has the advantage that the clusters will be of comparable compactness and that spurious solutions containing small components with very large concentration parameters are eliminated. In the following we derive how the M-step needs to be modified if the concentration parameters are restricted to be the same over components.

From Appendix A.2 of Banerjee *et al.* (2005) the optimal unconstrained κ_k can be obtained by solving

$$A_d(\kappa_k) = \frac{\| \sum_i p(k|x_i, \Theta) x_i \|}{\sum_i p(k|x_i, \Theta)}.$$

If the κ_k are constrained to be equal (but are not given), the optimal common κ can be obtained as follows. Using Equation A.12 in Banerjee *et al.* (2005), the modified Lagrangian becomes

$$\begin{aligned}
&\sum_k \left(\sum_i \log(C_d(\kappa)) p(k|x_i, \Theta) + \sum_i \kappa \mu_k^\top x_i p(k|x_i, \Theta) \right) + \lambda_k (1 - \mu_k^\top \mu_k) \\
&= \log(C_d(\kappa)) \sum_{k,i} p(k|x_i, \Theta) + \kappa \sum_{k,i} \mu_k^\top x_i p(k|x_i, \Theta) + \lambda_k (1 - \mu_k^\top \mu_k) \\
&= n \log(C_d(\kappa)) + \kappa \sum_{k,i} \mu_k^\top x_i p(k|x_i, \Theta) + \lambda_k (1 - \mu_k^\top \mu_k).
\end{aligned}$$

Setting partials with respect to μ_k and λ_k to zero as in the reference gives (again)

$$\mu_k = \frac{\sum_i x_i p(k|x_i, \Theta)}{\| \sum_i x_i p(k|x_i, \Theta) \|}$$

and with these μ_k we obtain for κ that

$$0 = -A_d(\kappa) n + \sum_{k,i} \mu_k^\top x_i p(k|x_i, \Theta) = -A_d(\kappa) n + \sum_k \left\| \sum_i x_i p(k|x_i, \Theta) \right\|,$$

i.e., κ needs to solve the equation

$$A_d(\kappa) = \frac{1}{n} \sum_k \left\| \sum_i x_i p(k|x_i, \Theta) \right\|.$$

3.3. Illustrative example: Household expenses

In the following the gender information is not used and it is investigated if finite mixtures allow to unravel a distinction between male and female respondents in their household expenses. Assuming that it is known that there are two underlying unobserved groups, a mixture with two components is fitted. The results are visualized in Figure 1 at the bottom left. The colors are according to assignments to components using the maximum a-posteriori probabilities. These assignments lead to one misclassification, i.e., one female is assigned to the component with the higher dispersion. The estimated parameters and the BIC value of the model are given in Table 1.

If the number of components is assumed to be unknown, the Bayesian information criterion (BIC) can be used to select a suitable number of components (see for example McLachlan

	π	housing	food	service	κ	BIC
$K = 2$	0.47	0.95	0.13	0.27	114.70	−200.34
	0.53	0.67	0.63	0.40	17.96	
$K = 3$	0.52	0.95	0.15	0.27	83.26	−211.55
	0.13	0.67	0.31	0.68	181.21	
	0.35	0.59	0.76	0.28	62.91	

Table 1: Results of fitting mixtures of vMF distributions to the household expenses example.

and Peel 2000, Chapter 6). In this case the minimum BIC is obtained for three components if models in the range of $K = 1, \ldots, 5$ are considered. The results of the mixture with $K = 3$ components are visualized in Figure 1 at the bottom right and the estimated parameters and the BIC value are given in Table 1. In this case the male respondents are split into two groups with less dispersion each and different mean directions. The R code for reproducing these results is provided in Section 4.3 after introducing package **movMF**.

4. Software

4.1. Main fitting function `movMF()`

The main function in package **movMF** for fitting mixtures of vMF distributions is `movMF()`, with synopsis

```
movMF(x, k, control = list(), ...)
```

The arguments for this function are as follows.

`x`: A numeric data matrix, with rows corresponding to observations. If necessary the data is standardized to unit row lengths. Furthermore, the matrix can be either stored as a dense matrix, a simple triplet matrix (defined in package **slam**, Hornik, Meyer, and Buchta 2014b), or a general sparse triplet matrix of class '`dgtMatrix`' (from package **Matrix**, Bates and Maechler 2014).

`k`: An integer indicating the number of components.

`control`: A list of control parameters consisting of

`E`: Specifies the variant of the EM algorithm used with possible values `"softmax"` (default), `"hardmax"` (classification EM), and `"stochmax"` (stochastic EM).

`kappa`: This argument allows to specify how to determine the concentration parameters.

- If numbers are given, the concentration parameters are assumed to be fixed and are not estimated in the EM algorithm.
- The method for solving for the concentration parameters can be specified by one of `"Banerjee_et_al_2005"`, `"Tanabe_et_al_2007"`, `"Sra_2012"`, `"Song_et_al_2012"`, `"uniroot"`, `"Newton"`, `"Halley"`, `"hybrid"` and `"Newton_Fourier"` (default). For more details see Section 2.2.
- For common concentration parameters a list with elements `common = TRUE` and a character string giving the estimation method needs to be provided.

`converge`: Logical indicating if convergence of the algorithm should be checked and if in this case the algorithm should be stopped before the maximum number of iterations is reached. For `E` equal to `"softmax"` this argument is set by default to `FALSE`. Note that only condition (a) of the convergence check (see Section 3.2) is assessed, i.e., the relative change in the log-likelihood values.

`maxiter`: Integer indicating the maximum number of iterations of the EM algorithm. (Default: 100.)

`reltol`: If the relative change in the log-likelihood falls below this threshold the EM algorithm is stopped if `converge` is `TRUE`. (Default: `sqrt(.Machine$double.eps)`.)

`verbose`: Logical indicating if information on the progress of the fitting process shall be printed during the estimation.

`ids`: Indicates either the class memberships of the observations or if equal to `TRUE` the class memberships are obtained from the attributes of the data. In this way the class memberships are for example stored if data is generated using function `rmovMF()`. If this argument is specified, the EM algorithm is stopped after one iteration, i.e., the parameter estimates are determined conditional on the known true class memberships.

`start`: Allows to specify the starting values used for initializing the EM algorithm which then starts with an M-step. It can either be a list of matrices where each matrix contains the a-posteriori probabilities of the observations or a list of vectors containing component assignments for the observations. Alternatively it can be a character vector with entries `"i"`, `"p"`, `"S"` or `"s"`. The length of the vector specifies how many different initializations are made. `"i"` indicates to randomly assign component memberships to the observations. The latter three draw observations as prototypes and determine a-posteriori probabilities by taking the implied cosine dissimilarities between observations and prototypes. `"p"` randomly picks observations as prototypes, `"S"` takes the first prototype to minimize the total cosine dissimilarity to the observations, and then successively picks observations farthest away from the already picked prototypes. For `"s"` one takes a randomly chosen observation as the first prototype, and then proceeds as for `"S"`. For more details on these initialization methods see package **skmeans** (Hornik *et al.* 2012; Hornik, Feinerer, and Kober 2014a) which uses the same initialization schemes.

`nruns`: An integer indicating the number of repeated runs of the EM algorithm with random initialization. This argument is ignored if either `ids` or `start` are specified.

`minalpha`: Components with size below the threshold indicated by `minalpha` are omitted during the estimation with the EM algorithm. This avoids estimation problems in the M-step if only very few observations are assigned to one component. The disadvantage is that the initial number of components is not necessarily equal to the number of components of the returned model.

4.2. Additional functionality in movMF

The object returned by `movMF()` has an S3 class called 'movMF' with methods `print()`, `coef()`, `logLik()` and `predict()` (yields either the component assignments or the matrix of a-posteriori probabilities). Additional functionality available in the package includes `rmovMF()` for drawing from a mixture of vMF distributions and `dmovMF()` for evaluating the mixture density.

4.3. Illustrative example: Household expenses

In the following the code for reproducing the results presented in Section 3.3 is provided. After loading the data set the three columns of interest are selected from the expenses and

stored in variable `x`. Then the classification variable `gender` is also extracted. First `movMF()` is used to fit a single vMF distribution to the two separate sub-samples and then mixtures are fitted with number of components varying from 1 to 5. To avoid reporting sub-optimal solutions where the EM algorithm was trapped in a local optimum, the best result from 20 random initializations is returned.

```
R> data("household", package = "HSAUR2")
R> x <- as.matrix(household[, c(1:2, 4)])
R> gender <- household$gender
R> theta <- rbind(female = movMF(x[gender == "female", ], k = 1)$theta,
+    male = movMF(x[gender == "male", ], k = 1)$theta)
R> set.seed(2008)
R> vMFs <- lapply(1:5, function(K)
+    movMF(x, k = K, control= list(nruns = 20)))
```

The BIC values for the different mixtures can be compared using

```
R> sapply(vMFs, BIC)

[1] -169.4291 -200.3364 -211.5490 -206.9498 -202.4944
```

5. Numerical issues

In what follows it will be convenient to write

$$H_\nu(\kappa) = {}_0F_1(;\nu+1;\kappa^2/4) = \frac{\Gamma(\nu+1)}{(\kappa/2)^\nu} I_\nu(\kappa).$$

As shown in Section 2, computing log-likelihoods for (mixtures of) vMF distributions on $\mathbb{R}^d$ requires the computation of $\log(H_{d/2-1})$, and ML estimation of concentration parameters amounts to solving equations of the form $A_d(\kappa) = \rho$, where

$$A_d(\kappa) = \frac{I_{d/2}(\kappa)}{I_{d/2-1}(\kappa)} = \frac{\kappa}{d}\frac{H_{d/2}(\kappa)}{H_{d/2-1}(\kappa)}$$

is the logarithmic derivative of $H_{d/2-1}$.

As $H_\nu(\kappa) \to \infty$ as $\kappa \to \infty$ (in fact, quite rapidly, see below), it clearly is a bad idea to try to compute $\log(H)$ as the logarithm of H, or A as the ratio of H functions. Similar considerations apply for using logarithms or ratios of incomplete modified Bessel functions I. One might wonder whether one could successfully take advantage of the fact that the Bessel functions provided by R are based on the **SPECFUN** package of Cody (1993) and hence also provide the exponentially scaled modified Bessel function $e^{-\kappa} I_\nu(\kappa)$ (the scaling is motivated by the asymptotic expansion $I_\nu(\kappa) \sim e^\kappa (2\pi\kappa)^{-1/2} \sum_m \alpha_m(\nu)/\kappa^m$ for $\kappa \to \infty$). However, exponential scaling does not help in situations where $1 \ll \kappa \ll \nu$: e.g., for $\kappa = 6000$ and $\nu = 10000$, H overflows whereas I underflows (even though R gives `Inf` and `0` for the cases without and with exponential scaling, respectively).

5.1. Computing A_d

One can use the Gauss continued fraction

$$\frac{I_\nu(z)}{I_{\nu-1}(z)} = \frac{1}{2\nu/z +} \frac{1}{2(\nu+1)/z +} \frac{1}{2(\nu+2)/z +} \cdots$$

for the ratio of modified Bessel functions (http://dlmf.nist.gov/10.33.E1) to compute A as

$$A_d(\kappa) = \frac{I_{d/2}(\kappa)}{I_{d/2-1}(\kappa)} = \frac{1}{d/\kappa +} \frac{1}{(d+2)/\kappa +} \frac{1}{(d+4)/\kappa +} \cdots$$

(Equation 4.3 in Banerjee *et al.* 2005), using, e.g., Steed's method (e.g., http://dlmf.nist.gov/3.10) for evaluation. However, as pointed out by Gautschi and Slavik (1978) and Tretter and Walster (1980) (and, quite recently, re-iterated by Song *et al.* 2012), the Perron continued fraction

$$\frac{I_\nu(z)}{I_{\nu-1}(z)} = \frac{z}{2\nu + z -} \frac{(2\nu+1)z}{2\nu + 1 + 2z -} \frac{(2\nu+3)z}{2\nu + 2 + 2z -} \frac{(2\nu+5)z}{2\nu + 3 + 2z -} \cdots$$

is numerically more stable (as computing it by forward recursion only accumulates positive terms, whereas for the Gauss continued fraction the terms alternate in sign), and converges substantially faster for positive $z \gg \nu$. Hence, by default we compute A via the Perron continued fraction (with implementation based on Equation 3.3′ in Gautschi and Slavik 1978), and additionally provide computation via the Gauss continued fraction (with implementation based on Equation 3.2′ in Gautschi and Slavik 1978) and using exponentiation of $\log(H)$ differences as alternatives.

For κ close to zero it is better (and necessary for $\kappa = 0$) to use the approximation

$$A_d(\kappa) = \frac{1}{d}\kappa - \frac{1}{d^2(d+2)}\kappa^3 + O(\kappa^5), \qquad \kappa \to 0$$

(Schou 1978, Equation 5). The $O(\kappa^5)$ can be made more precise by using the series representation $I_\nu(z) = \sum_{n=0}^{\infty}(z/2)^{2n+\nu}/(n!\Gamma(n+\nu+1))$ so that for $\kappa \to 0$,

$$\begin{aligned} A_d(\kappa) &= \frac{(\kappa/2)^{d/2}\left(\frac{1}{\Gamma(d/2+1)} + \frac{\kappa^2/4}{\Gamma(d/2+2)} + \frac{\kappa^4/32}{\Gamma(d/2+3)} + O(\kappa^6)\right)}{(\kappa/2)^{d/2-1}\left(\frac{1}{\Gamma(d/2)} + \frac{\kappa^2/4}{\Gamma(d/2+1)} + \frac{\kappa^2/32}{\Gamma(d/2+2)} + O(\kappa^6)\right)} \\ &= \frac{\kappa}{2}\frac{\frac{2}{d} + \frac{\kappa^2}{d(d+2)} + \frac{\kappa^4}{4d(d+2)(d+4)} + O(\kappa^6)}{1 + \frac{\kappa^2}{2d} + \frac{\kappa^4}{8d(d+2)} + O(\kappa^6)} \end{aligned}$$

from which the coefficient of κ^5 can straightforwardly be determined as

$$\frac{1}{2}\left(\frac{1}{4d(d+2)(d+4)} - \frac{1}{2d^2(d+2)} - \frac{2}{8d^2(d+2)} + \frac{2}{4d^3}\right) = \frac{2}{d^3(d+2)(d+4)}$$

so that

$$A_d(\kappa) = \frac{1}{d}\kappa - \frac{1}{d^2(d+2)}\kappa^3 + \frac{2}{d^3(d+2)(d+4)}\kappa^5 + O(\kappa^7), \qquad \kappa \to 0.$$

From this approximations for A' and A'' can be obtained by term-wise differentiation and also used for κ close to 0.

5.2. Computing $\log(H_\nu)$

Write

$$R_\nu(\kappa) = \frac{I_{\nu+1}(\kappa)}{I_\nu(\kappa)}$$

for the Bessel function ratio (so that $A_d = R_{d/2-1}$) and

$$G_{\alpha,\beta}(\kappa) = \frac{\kappa}{\alpha + \sqrt{\kappa^2 + \beta^2}}.$$

Amos (1974, Equations 11 and 16) shows that for all non-negative κ and ν, $G_{\nu+1/2,\nu+3/2}(\kappa) \leq R_\nu(\kappa) \leq G_{\nu,\nu+2}(\kappa)$. With $\beta_{SS}(\nu) = \sqrt{(\nu+1/2)(\nu+3/2)}$, Theorem 2 of Simpson and Spector (1984) implies the upper bound $R_\nu(\kappa) \leq G_{\nu+1/2,\beta_{SS}(\nu)}(\kappa)$ for all non-negative κ and ν. We thus have

$$G_{\nu+1/2,\nu+3/2}(\kappa) \leq R_\nu(\kappa) \leq \min\left(G_{\nu,\nu+2}(\kappa), G_{\nu+1/2,\beta_{SS}(\nu)}(\kappa)\right) \tag{4}$$

(from which the bounds of Equation 3 are obtained by inversion). In addition, using results in Schou (1978, Equations 5 and 6), one can show that $G_{\nu,\nu+2}$ and $G_{\nu+1/2,\beta_{SS}(\nu)}$ are second order exact approximations for $\kappa \to 0$ and $\kappa \to \infty$, respectively (Hornik and Grün 2013).

As $\log(H_\nu)' = R_\nu$ (and $H_\nu(0) = 1$), integration gives

$$\log(H_\nu(\kappa)) = \int_0^\kappa R_\nu(t)\, dt.$$

Thus, the bounds for the Bessel function ratio R_ν can be used to obtain bounds for H_ν. Writing

$$S_{\alpha,\beta}(\kappa) = \sqrt{\kappa^2 + \beta^2} - \alpha \log(\alpha + \sqrt{\kappa^2 + \beta^2}) - \beta + \alpha \log(\alpha + \beta),$$

it is easily verified that $S'_{\alpha,\beta} = G_{\alpha,\beta}$ and $S_{\alpha,\beta}(0) = 0$. Using the Amos-type bounds from Equation 4, we thus obtain that for $\nu \geq 0$ and $\kappa \geq 0$,

$$S_{\nu+1/2,\nu+3/2}(\kappa) \leq \log(H_\nu(\kappa)) \leq \min\left(S_{\nu,\nu+2}(\kappa), S_{\nu+1/2,\beta_{SS}(\nu)}(\kappa)\right). \tag{5}$$

Where a single approximating value is sought, we prefer to use the upper bound which is based on the combination of upper Amos-type bounds which are second order exact at zero and infinity. Again, the bounds in Equation 5 are surprisingly tight.

Result. *Let*

$$s(\nu) = (\nu + 3/2) - \beta_{SS}(\nu) - (\nu + 1/2) \log \frac{2(\nu+1)}{\nu + 1/2 + \beta_{SS}(\nu)},$$

with $\nu \geq 0$.

Then s *is non-increasing on* $[0, \infty)$ *with* $s(0) = (3 - \sqrt{3} + \log((1+\sqrt{3})/4))/2 = 0.4433537$ *and* $\lim_{\nu\to\infty} s(\nu) = 1/4$. *For* $\nu_0 \geq 0$,

$$\sup_{\kappa\geq 0, \nu \geq \nu_0} (S_{\nu+1/2,\beta_{SS}(\nu)}(\kappa) - S_{\nu+1/2,\nu+3/2}(\kappa)) = s(\nu_0).$$

For $\beta_{SS}(\nu) \leq \beta \leq \nu + 3/2$,

$$\sup_{\kappa \geq 0} |\log(H_\nu(\kappa)) - S_{\nu+1/2,\beta}(\kappa)| \leq s(\nu).$$

Proof. For simplicity, write $\alpha = \nu + 1/2$, $\beta_L = \nu + 3/2$ and $\beta_U = \beta_{SS}(\nu)$, omitting the dependence on ν. We have $\beta_U \leq \beta_L$ and $G_{\alpha,\beta_U} \geq G_{\alpha,\beta_L}$. Hence,

$$S_{\alpha,\beta_U}(\kappa) - S_{\alpha,\beta_L}(\kappa) = \int_0^\kappa (G_{\alpha,\beta_U}(t) - G_{\alpha,\beta_L}(t))\, dt$$

is non-decreasing in κ, and attains its supremum for $\kappa \to \infty$. Now,

$$\begin{aligned} & S_{\alpha,\beta_U}(\kappa) - S_{\alpha,\beta_L}(\kappa) \\ = \;& \sqrt{\kappa^2 + \beta_U^2} - \sqrt{\kappa^2 + \beta_L^2} - \alpha \log \left(\frac{\alpha + \sqrt{\kappa^2 + \beta_U^2}}{\alpha + \sqrt{\kappa^2 + \beta_L^2}} \right) + (\beta_L - \beta_U) - \alpha \log \frac{\alpha + \beta_L}{\alpha + \beta_U}. \end{aligned}$$

As

$$\sqrt{\kappa^2 + \beta_U^2} - \sqrt{\kappa^2 + \beta_L^2} = \frac{(\kappa^2 + \beta_U^2) - (\kappa^2 + \beta_L^2)}{\sqrt{\kappa^2 + \beta_U^2} + \sqrt{\kappa^2 + \beta_L^2}} = \frac{\beta_U^2 - \beta_L^2}{\sqrt{\kappa^2 + \beta_U^2} + \sqrt{\kappa^2 + \beta_L^2}},$$

we have

$$\lim_{\kappa \to \infty} (S_{\alpha,\beta_U}(\kappa) - S_{\alpha,\beta_L}(\kappa)) = (\beta_L - \beta_U) - \alpha \log \frac{\alpha + \beta_L}{\alpha + \beta_U} = s(\nu).$$

The value of $s(0)$ is obtained by insertion, and $\lim_{\nu \to \infty} s(\nu)$ can be obtained as follows. We have

$$\beta_L - \beta_U = \sqrt{\alpha + 1}(\sqrt{\alpha + 1} - \sqrt{\alpha}) = \frac{\sqrt{\alpha + 1}}{\sqrt{\alpha + 1} + \sqrt{\alpha}} \to 1/2$$

as $\nu \to \infty$, and

$$\begin{aligned} \frac{\alpha + \beta_U}{\alpha + \beta_L} &= \frac{\alpha + \sqrt{\alpha(\alpha + 1)}}{2\alpha + 1} \\ &= \frac{(1 + \sqrt{1 + 1/\alpha})/2}{1 + 1/(2\alpha)} \\ &= \frac{1}{2} \left(1 + 1 + \frac{1}{2\alpha} + O(\alpha^{-2}) \right) \left(1 - \frac{1}{2\alpha} + O(\alpha^{-2}) \right) \\ &= 1 - \frac{1}{4\alpha} + O(\alpha^{-2}) \end{aligned}$$

so that

$$\alpha \log \frac{\alpha + \beta_U}{\alpha + \beta_L} = \alpha \left(-\frac{1}{4\alpha} + O(\alpha^{-2}) \right) = -\frac{1}{4} + O(\alpha^{-1}) \to -1/4$$

as $\nu \to \infty$. Hence, $\lim_{\nu \to \infty} s(\nu) = 1/4$.

To show that s is non-increasing, note that we have

$$\alpha + \beta_L = 2\alpha + 1, \qquad \frac{d\alpha}{d\nu} = \frac{d\beta_L}{d\nu} = 1, \qquad \frac{d\beta_U}{d\nu} = \frac{2\alpha + 1}{2\beta_U}$$

and hence

$$\frac{ds}{d\nu} = 1 - \frac{2\alpha+1}{2\beta_U} - \log\frac{2\alpha+1}{\alpha+\beta_U} - \alpha\left(\frac{2}{2\alpha+1} - \frac{1}{\alpha+\beta_U}\left(1+\frac{2\alpha+1}{2\beta_U}\right)\right).$$

For non-negative t and τ,

$$\log\frac{t+\tau}{t} = \int_0^\tau \frac{1}{t+s}\,ds \geq \frac{1}{t+\tau}\int_0^\tau ds = \frac{\tau}{t+\tau}.$$

Hence, with $t = \alpha + \beta_U$ and $\tau = \beta_L - \beta_U$,

$$\log\frac{2\alpha+1}{\alpha+\beta_U} \geq \frac{\beta_L - \beta_U}{2\alpha+1}$$

and

$$\begin{aligned}
\frac{ds}{d\nu} &\leq 1 - \frac{2\alpha+1}{2\beta_U} + \frac{\beta_U-\beta_L}{2\alpha+1} + \frac{\alpha}{\alpha+\beta_U}\left(1+\frac{2\alpha+1}{2\beta_U}\right) - \frac{2\alpha}{2\alpha+1} \\
&= \frac{2\alpha+1}{2\beta_U}\left(\frac{\alpha}{\alpha+\beta_U} - 1\right) + \frac{\alpha}{\alpha+\beta_U} + \frac{2\alpha+1+\beta_U-\beta_L-2\alpha}{2\alpha+1} \\
&= -\frac{2\alpha+1}{2(\alpha+\beta_U)} + \frac{\alpha}{\alpha+\beta_U} + \frac{\beta_U-\alpha}{2\alpha+1} \\
&= -\frac{1}{2(\alpha+\beta_U)} + \frac{\beta_U-\alpha}{2\alpha+1} \\
&= \frac{-(2\alpha+1)+2(\beta_U^2-\alpha^2)}{2(\alpha+\beta_U)(2\alpha+1)} \\
&= \frac{-1}{2(\alpha+\beta_U)(2\alpha+1)} \\
&\leq 0,
\end{aligned}$$

establishing that s is non-increasing.

Finally, for $\beta_{SS}(\nu) \leq \beta \leq \nu + 3/2$, both $\log(H_\nu)$ and $S_{\nu+1/2,\beta}$ are bounded below by $S_{\nu+1/2,\nu+3/2}$ and above by $S_{\nu+1/2,\beta_{SS}(\nu)}$, so that $|\log(H_\nu) - S_{\nu+1/2,\beta}| \leq (S_{\nu+1/2,\beta_{SS}(\nu)} - S_{\nu+1/2,\nu+3/2}) \leq s(\nu)$, completing the proof. □

These results can be used to derive the following approach to computing $\log(H_\nu)$. Choose a threshold θ such that e^θ does not overflow and $e^{-\theta}$ does not underflow. Using IEEE 754 double precision floating point computations, we can take $\theta = 700$. Choose an approximation L_ν for $\log(H_\nu)$ for which $S_{\nu+1/2,\nu+3/2} \leq L_\nu \leq S_{\nu+1/2,\beta_{SS}(\nu)}$ for all $\kappa \geq 0$. By the above, this has approximation error at most $s(0) < 1/2$. Possible choices for L_ν are convex combinations of the lower and upper bounds in Equation 5 (e.g., simply take the upper bound) or $S_{\nu+1/2,\beta}(\kappa)$ with some $\beta_{SS}(\nu) \leq \beta \leq \nu + 3/2$ (e.g, $\beta = \nu + 1$); in the package we use

$$\begin{aligned}
L_\nu(\kappa) &= \int_0^\kappa \min(G_{\nu,\nu+2}(t), G_{\nu+1/2,\beta_{SS}(\nu)}(t))\,dt \\
&= S_{\nu+1/2,\beta_{SS}(\nu)}(\kappa) + (S_{\nu,\nu+2}(\min(\kappa,\kappa_\nu)) - S_{\nu+1/2,\beta_{SS}(\nu)}(\min(\kappa,\kappa_\nu))),
\end{aligned}$$

where $\kappa_\nu = \sqrt{(3\nu + 11/2)(\nu + 3/2)}$ is the positive root of $G_{\nu,\nu+2}(\kappa) = G_{\nu+1/2,\beta_{SS}(\nu)}(\kappa)$. Then if $L_\nu(\kappa) \leq \theta - 1/2$ (so that $\log(H_\nu(\kappa)) \leq \theta$), compute $H_\nu(\kappa)$ by its series expansion, and take the logarithm of this. If $\theta - 1/2 < L_\nu(\kappa) \leq 2\theta - 1$ so that

$$|\log(H_\nu(\kappa)) - L_\nu(\kappa)/2| \leq |\log(H_\nu(\kappa)) - L_\nu(\kappa)| + L_\nu(\kappa)/2 \leq \theta,$$

and hence $e^{-L_\nu(\kappa)/2} H_\nu(\kappa)$ does not over- or underflow, write

$$H_\nu(\kappa) = e^{L_\nu(\kappa)/2} e^{-L_\nu(\kappa)/2} H_\nu(\kappa) = e^{L_\nu(\kappa)/2} \sum_{m=0}^{\infty} \frac{e^{-L_\nu(\kappa)/2}\Gamma(\nu+1)}{\Gamma(\nu+1+m)} \frac{(\kappa^2/4)^m}{m!},$$

and compute $\log(H_\nu(\kappa))$ as $L_\nu(\kappa)/2$ plus the logarithm of the sum of the scaled series. Otherwise, use the approximation $L_\nu(\kappa)$ (note that with $\theta = 700$, this has a relative approximation error of about at most $1/2 \cdot 1/1400 \leq 0.0004$). This is the approach for computing $\log(H_\nu)$ currently implemented in package **movMF**.

Alternatively, one can use the bounds to obtain refined strategies of computing $\log(H_\nu)$ using available codes for modified Bessel functions. We have

$$\log(H_\nu(\kappa)) = \log(I_\nu(\kappa)) - \nu \log(\kappa/2) + \log(\Gamma(\nu+1))$$

and hence, using Stirling's approximation $\log(\Gamma(z)) \approx (z - 1/2)\log(z) - z + \log(2\pi)/2$ and $L_\nu = S_{\nu+1/2,\nu+1}$ for notational convenience, gives

$$\begin{aligned}
&\log(I_\nu(\kappa)) \\
&\quad = \log(H_\nu(\kappa)) + \nu \log(\kappa/2) - \log(\Gamma(\nu+1)) \\
&\quad \approx L_\nu(\kappa) + \nu \log(\kappa/2) - (\nu+1/2)\log(\nu+1) + (\nu+1) - \frac{\log(2\pi)}{2} \\
&\quad = \sqrt{\kappa^2 + (\nu+1)^2} + (\nu+1/2)\log\frac{\kappa}{\nu+1/2+\sqrt{\kappa^2+(\nu+1)^2}} \\
&\qquad - \frac{\log(\kappa/2)}{2} + (\nu+1/2)\log\frac{2\nu+3/2}{2(\nu+1)} - \frac{\log(2\pi)}{2}.
\end{aligned}$$

This implies

$$\log(I_\nu(\kappa)) = \sqrt{\kappa^2 + (\nu+1)^2} + (\nu+1/2)\log\frac{\kappa}{\nu+1/2+\sqrt{\kappa^2+(\nu+1)^2}} - \frac{\log(\kappa)}{2} + O(1),$$

where the $O(1)$ can be made more precise, giving a first-order variant of the large ν uniform asymptotic approximation for I_ν given by Olver (1954). (For $\nu \to \infty$, the error in the approximation for $\log(\Gamma)$ tends to zero, and as $(\nu+1/2)\log((2\nu+3/2)/(2\nu+2)) \to -1/4$, the $O(1)$ becomes $-\log(\pi)/2 - 1/4 + o(1)$ (modulo the error in the approximation of $\log(H_\nu(\kappa))$ by $L_\nu(\kappa)$ which is at most $1/4 + o(1)$).

From the above, we can see that when $\kappa \gg \nu$, $I_\nu(\kappa)$ overflows quite rapidly; on the other hand, for $\kappa = o(\nu)$ and $\nu \to \infty$, $I_\nu(\kappa)$ underflows quite rapidly. For computing logarithms, overflow can be avoided by employing the exponentially scaled modified Bessel function $e^{-\kappa} I_\nu(\kappa)$ (as commonly available in codes for computing Bessel functions, such as the **SPECFUN** package Cody 1993, used by R) and computing $\log(I_\nu(\kappa)) = \kappa + \log(e^{-\kappa} I_\nu(\kappa))$. However, this clearly does not help avoiding underflow.

This motivates the following strategy for computing $\log(I_\nu(\kappa))$ for a wide range of values. Start by computing a quick approximation $T_\nu(\kappa)$ to $\log(I_\nu)$, either using the above $L_\nu(\kappa) + \nu\log(\kappa/2) - \log(\Gamma(\nu+1))$ or the leading term of the large ν uniform asymptotic approximation (in R, the latter is available via function `besselI.nuAsym()` in package **Bessel**; Maechler 2013). If this is "sufficiently small" in absolute value (so that $I_\nu(\kappa)$ will neither over- nor underflow), compute $\log(I_\nu(\kappa))$ directly as the logarithm of $I_\nu(\kappa)$. Otherwise, if the approximation is "too large", but $T_\nu(\kappa) - \kappa$ is sufficiently small in absolute value so that the exponentially scaled $e^{-\kappa}I_\nu(\kappa)$ can be computed without over- or underflow, compute $\log(I_\nu(\kappa))$ as $\kappa + \log(e^{-\kappa}I_\nu(\kappa))$. Otherwise, use the quick approximation, or the large ν uniform asymptotic approximation with additional terms (package **Bessel** allows up to 4 additional terms). This strategy can readily be translated into a strategy for computing $\log(H_\nu)$.

6. Comparison to spherical k-means

Finite mixtures of vMF distributions are the model-based clustering counterpart for spherical k-means which is a distance-based clustering approach. This is analogous to finite mixtures of multivariate Gaussian distributions being the model-based clustering counterpart for k-means clustering. In the following the performance of finite mixtures of vMF distributions is compared to spherical k-means on a standard set of text mining corpora and artificial data where the concentration parameters as well as the component sizes are either restricted to be equal over components or allowed to differ in order to illustrate situations where the mixture modeling approach is preferable.

6.1. Using text mining corpora with known classifications

The software package **CLUTO** (Karypis 2003) is a freely available, but closed-source software package for clustering low- and high-dimensional data sets. According to its documentation it is well-suited for clustering data sets arising in many diverse application areas including information retrieval, customer purchasing transactions, web, GIS, science, and biology. The package provides spherical k-means clustering, i.e., k-means clustering where instead of minimizing the squared Euclidean distance between observations and their assigned prototypes the cosine similarity between the observations and their prototypes is maximized. The software was evaluated in Zhao and Karypis (2004) using several data sets derived from text corpora. Slightly different versions of these data sets are available at `http://www.cs.umn.edu/~karypis/cluto/files/datasets.tar.gz`. An overview of the general characteristics of the data sets is given in Table 2. The sources of the data sets are outlined in Zhao and Karypis (2004) as well as the pre-processing steps which consisted of removing common words listed in a stop-list and stemming words using Porter's suffix-stripping algorithm.

In the following the data is clustered using spherical k-means as well as the model-based alternative of finite mixtures of vMF distributions and the results are compared. For spherical k-means package **skmeans** is used. Prior to the analysis all terms occurring in less than two documents were removed as also suggested in Zhao and Karypis (2004).

As input the raw data document-term matrices are used where rows correspond to documents and columns to the terms. The raw data consisting of the term frequencies are used as well as transformed data, where instead of term frequency (TF) weighting term frequency-

Data set	# of documents	# of terms	# of classes
classic	7094	13576	4
fbis	2463	2000	17
hitech	2301	15377	6
k1a	2340	15077	20
k1b	2340	15077	6
la12	6279	23620	6
mm	2521	20759	2
new3	9558	43993	44
ohscal	11162	11465	10
re0	1504	2886	13
re1	1657	3758	25
reviews	4069	25418	5
sports	8580	19801	7
tr11	414	6429	9
tr12	313	5804	8
tr23	204	5832	6
tr31	927	10128	7
tr41	878	7454	10
tr45	690	8261	10
wap	1560	8460	20

Table 2: Summary of data sets used to compare mixtures of vMF distributions to spherical k-means clustering.

inverse document frequency (TF-IDF, Salton and Buckley 1988) weighting is applied. TF-IDF weighting has the advantage to downweight terms which are very frequent in the corpus. The entries of the document-term matrix for TF-IDF weighting are given by

$$\frac{t_{wd}}{\sum_{v=1}^{m} t_{vd}} \frac{n}{\sum_{f=1}^{n} \mathbf{1}_{t_{wf}>0}},$$

where t_{wd} denotes the frequency how often word w occurred in document d, n is the number of documents and m the size of the vocabulary.

A clustering of the data can be obtained from a fitted mixture model by assigning each observation to the component where the a-posteriori probability is maximum. We denote this assignment by the variable c_{ik} which is one if observation i is assigned to component k and zero otherwise. Furthermore we set $n_k = \sum_{i=1}^{n} c_{ik}$ to denote the cluster size for each k. These clustering results of the finite mixtures of vMF distributions are evaluated using entropy and purity as performance measures (Zhao and Karypis 2001, 2004) and the results are compared to those obtained using spherical k-means clustering.

Given a particular cluster S_k of size n_k, the entropy of this cluster is defined to be

$$E(S_k) = -\frac{1}{\log q} \sum_{i=1}^{q} \frac{n_k^i}{n_k} \log \frac{n_k^i}{n_k},$$

where q is the number of classes in the data set, and n_k^i is the number of documents of the ith class assigned to the kth cluster. The entropy of the entire clustering solution is then defined

by

$$\text{Entropy} = \sum_{k=1}^{K} \frac{n_k}{n} E(S_k).$$

Given a particular cluster S_k of size n_k, the purity of this cluster is defined to be

$$P(S_k) = \frac{1}{n_k} \max_i (n_k^i).$$

The purity of the entire clustering solution is then defined by

$$\text{Purity} = \sum_{k=1}^{K} \frac{n_k}{n} P(S_k).$$

Better clustering solutions are indicated by, respectively, smaller entropy and larger purity values.

A further criterion for evaluating cluster solutions is the normalized mutual information (NMI; Strehl and Ghosh 2002) given by

$$\text{NMI} = \frac{\sum_{k=1}^{K} \sum_{i=1}^{q} n_k^i \log \left(\frac{n \cdot n_k^i}{n_k \cdot n^i} \right)}{\sqrt{\left(\sum_{k=1}^{K} n_k \log \frac{n_k}{n} \right) \left(\sum_{i=1}^{q} n^i \log \frac{n^i}{n} \right)}},$$

where n^i is the number of documents in the ith class. The NMI criterion takes values in $[0, 1]$ and higher values indicate better clusterings. This criterion is available in package **clue** (Hornik 2005, 2014) through function `cl_agreement()` with `method = "NMI"`.

First, the results are compared when using the true categorization of the observations for initializing the algorithms. For spherical k-means method `"pclust"` is used to determine the final partition (for details on this and other available methods in **skmeans** see Hornik *et al.* 2012). For finite mixtures of vMF distributions common concentration parameters as well as different concentration parameters over components are fitted. In Figure 2 the results for the three criteria – entropy, purity and NMI – are given in aggregate form over all twenty data sets.

The figure indicates that the cluster solutions are better if the TF-IDF weighting instead of TF weighting is used regardless of fitting method or performance criterion used. Furthermore, the results are in general best for mixtures of vMF distributions with common concentration parameters followed by mixtures with different concentration parameters and spherical k-means clustering is worst.

In the second comparison the number of clusters is assumed to be known, but the class memberships are assumed to not be available for initialization. The algorithms are randomly initialized 20 times using the default initialization method and the best solution is retained. For spherical k-means clustering method `"CLUTO"` from package **skmeans** is used, which uses internally the stand-alone software **CLUTO** (Karypis 2006) by invoking it through the command-line interface. The results are given in Figure 3. In this case TF-IDF weighting

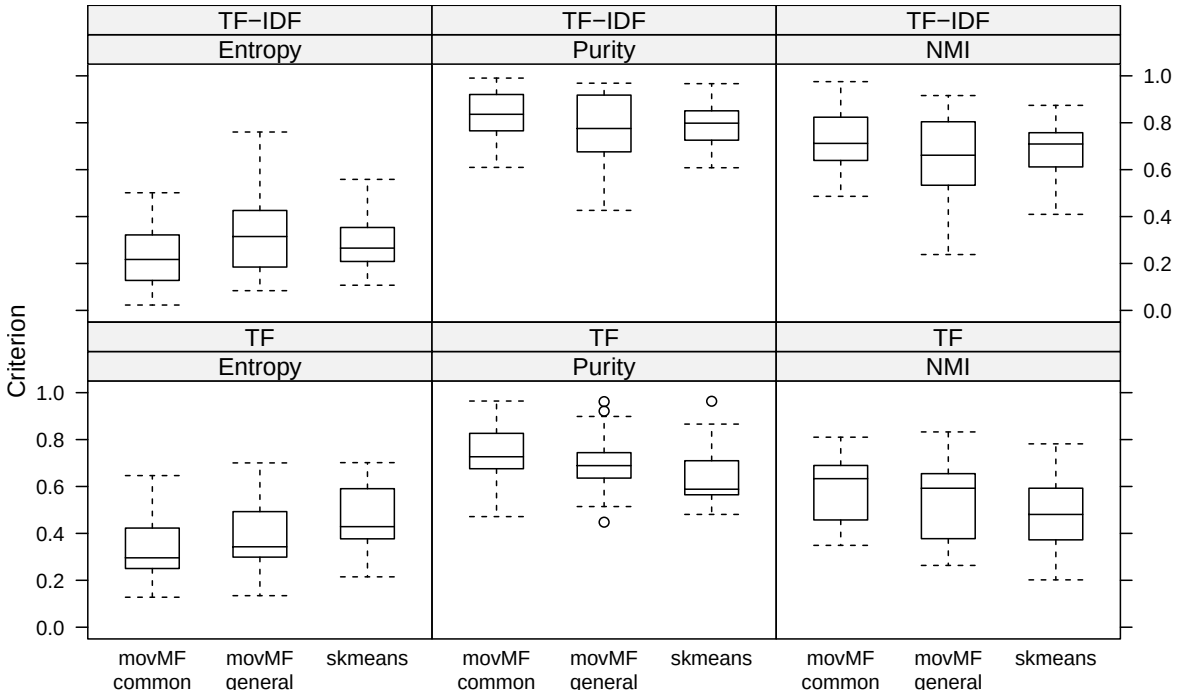

Figure 2: Comparison of results for mixtures of vMF distributions and spherical k-means for initialization using the known class memberships.

in general again leads to better results than TF weighting. Furthermore, spherical k-means clustering gives about equally good results than mixtures of vMF distributions with common concentration parameters. For mixtures of vMF distributions with general (not constrained to be equal) concentration parameters the results are always slightly inferior compared to the other two methods.

An investigation of the log-likelihood values of the fitted models indicates that in almost all cases (more than 95%) random initialization resulted in models with higher log-likelihood values than when using initialization in the true classification. Furthermore, the BIC values indicate that the models with general concentration parameters always have a superior fit than the constrained models. Hence, from a density estimation point of view the general model using random initialization gave the best results. However, with respect to classification performance this model did not give superior results in this application.

6.2. Using artificial data

In the following the results for artificial data are compared where (1) the component sizes are set to be equal or unequal and (2) the concentration parameters are the same or different over components. The performance of the three methods should be about the same if the component sizes are equal and if the concentration parameters are the same over components. The performance of spherical k-means should be worse in the case of different component sizes in comparison to the mixture model approach. Differences in the concentration parameters should deteriorate the performance of spherical k-means and mixtures of vMF distributions with common concentration parameters in comparison to the general mixture model method.

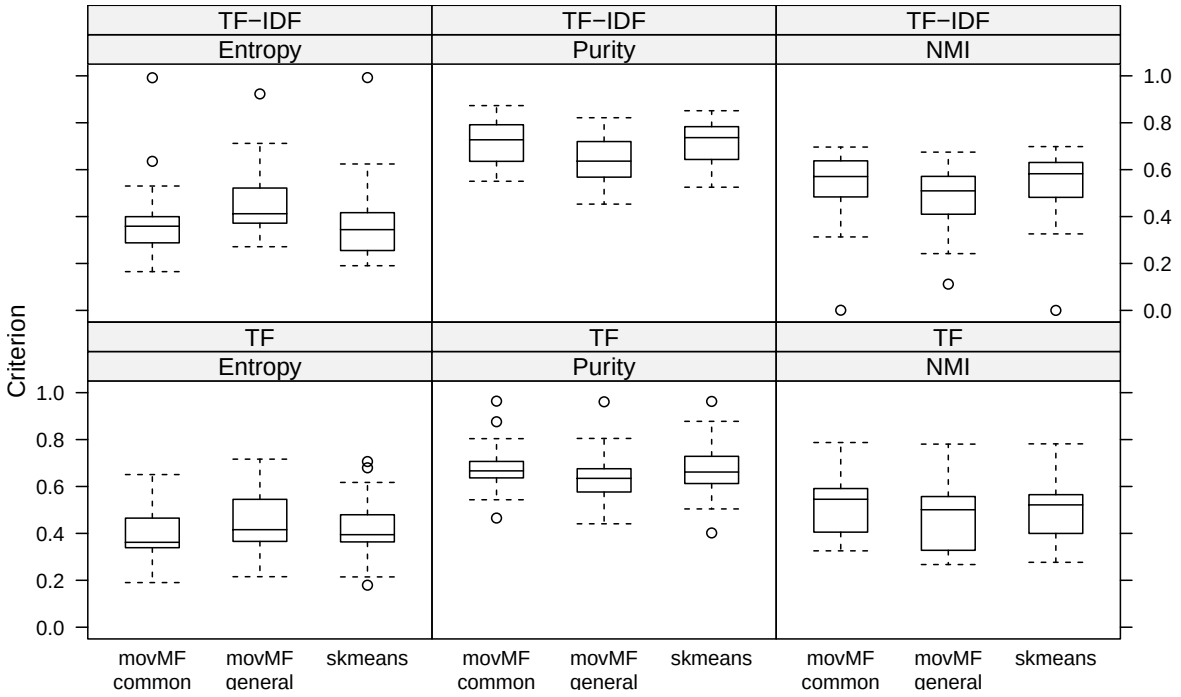

Figure 3: Comparison of results for mixtures of vMF distributions and spherical k-means for the best solution of 20 random initializations.

In the following we use mixtures with three components and data which is 300-dimensional. We use the following parameters for the component sizes and the concentration parameters. The μs are assumed to differ only for a subset of the variables over the components and are the same for each combination of component sizes and concentration parameter setting.

```
R> pis <- list(equal = prop.table(rep(1, 3)),
+    unequal = prop.table(c(1, 9, 10)))
R> kappas <- list(equal = rep(200, 3), unequal = c(10, 200, 100))
```

We sample 100 data sets from each of the mixture models and fit the different models to the data using 20 random initializations. The results are shown in Figure 4 using entropy as performance measure. For the other performance measures the results are comparable and therefore not shown. Clearly in the case where the components all have the same size and the same concentration parameters all methods have comparably good performance. In the case where the component sizes differ, but concentration parameters are the same spherical k-means performs considerable worse than the finite mixture approach. In the case where the concentration parameters differ over components only the finite mixture approach where different concentration parameters are fitted leads to a good performance while the other two approaches have a considerable worse performance, but perform similarly well.

7. Application: useR! 2008 abstracts

In 2008 the "useR! 2008", the 3rd international R user conference, took place in Dortmund, Germany. In total 177 abstracts were submitted and accepted for presentation at the confer-

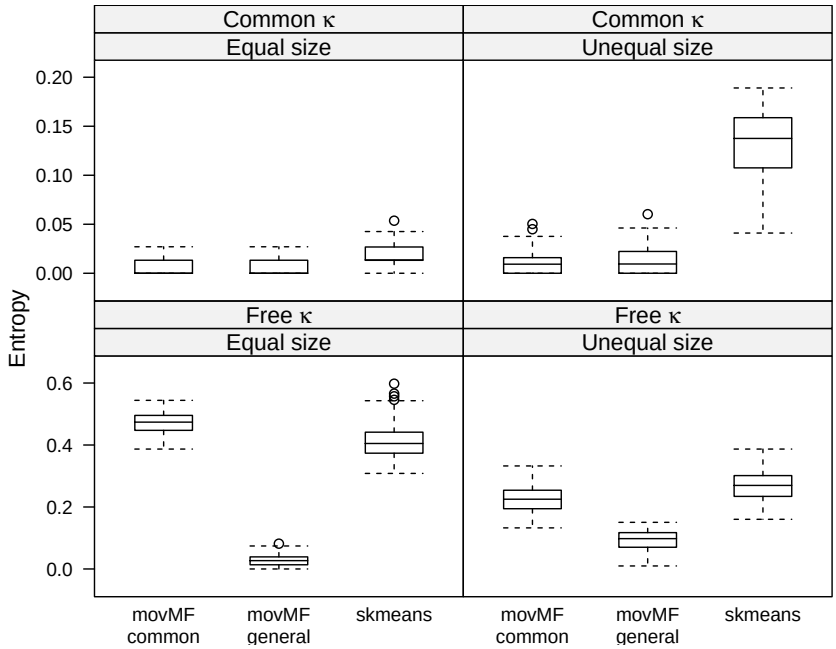

Figure 4: Comparison of the entropy values for mixtures of vMF distributions and spherical k-means for the artificial data.

ence. The abstracts with additional information such as title, author, session, and keywords are available in the R data package **corpus.useR.2008.abstracts** available from the repository at http://datacube.wu.ac.at/.

The following code checks if the package is installed and if necessary installs it. Furthermore, the data contained in the package is loaded.

```
R> if (!require("corpus.useR.2008.abstracts", quietly = TRUE)) {
+    install.packages("corpus.useR.2008.abstracts",
+      repos = "http://datacube.wu.ac.at/", type = "source")
+  }
R> data("useR_2008_abstracts", package = "corpus.useR.2008.abstracts")
```

Using the **tm** package (Feinerer, Hornik, and Meyer 2008; Feinerer and Hornik 2014) the data can be pre-processed by (1) generating a corpus from the vector of abstracts and (2) building a document-term matrix from the corpus. For constructing the document-term matrix each abstract needs to be tokenized (i.e., split into words, e.g., by using white space characters as separators) and transformed to lower case. Punctuation as well as numbers can be removed and the words can be stemmed (i.e., inflected words are reduced to a base form). In addition a minimum length can be imposed on the words as well as a minimum and maximum frequency within an abstract required.

The vector of abstracts is transformed to an object which has an extended class of '`Source`' using `VectorSource()`. This object is used as input for `Corpus()` to generate the corpus. The map from the corpus to the document-term matrix is performed using `DocumentTermMatrix()`. The `control` argument of `DocumentTermMatrix()` specifies which pre-processing steps are applied to determine the frequency vectors of term occurrences in each abstract. We use the

titles and the abstracts together to construct the document-term matrix.

```
R> library("tm")
R> abstracts_titles <- apply(useR_2008_abstracts[,c("Title", "Abstract")],
+      1, paste, collapse = " ")
R> useR_2008_abstracts_corpus <- Corpus(VectorSource(abstracts_titles))
R> useR_2008_abstracts_DTM <-
+    DocumentTermMatrix(useR_2008_abstracts_corpus,
+      control = list(tokenize = "MC", stopwords = TRUE, stemming = TRUE,
+        wordLengths = c(3, Inf)))
```

Method `"MC"` was used for tokenizing. This method aims at producing the same results as the **MC** toolkit for creating vector models from text documents (Dhillon and Modha 2001; Dhillon, Fan, and Guan 2001). In addition the words are stemmed, a set of stop words are removed and all words are kept which have a length of at least 3.

The resulting document-term matrix has 177 rows and 3853 columns. The ten most frequent terms (occurring in different abstracts) are the following.

```
R> library("slam")
R> ColSums <- col_sums(useR_2008_abstracts_DTM > 0)
R> sort(ColSums, decreasing = TRUE)[1:10]
```

```
     use   packag     data      can    model     base  develop
     150      127      121      112      103       91       91
 analysi   method function
      88       85       82
```

To reduce the dimension of the problem and omit terms which occur too frequently or too infrequently in the corpus to be of use when clustering the abstracts, we omit all terms which occur in less than 5 abstracts and more than 90 abstracts.

```
R> useR_2008_abstracts_DTM <-
+    useR_2008_abstracts_DTM[, ColSums >= 5 & ColSums <= 90]
R> useR_2008_abstracts_DTM
```

```
<<DocumentTermMatrix (documents: 177, terms: 860)>>
Non-/sparse entries: 12384/139836
Sparsity           : 92%
Maximal term length: 15
Weighting          : term frequency (tf)
```

The data is transformed using TF-IDF weighting.

```
R> useR_2008_abstracts_DTM <- weightTfIdf(useR_2008_abstracts_DTM)
```

In the following different mixtures of vMF distributions are fitted to training data using 10-fold cross-validation and are compared based on the predictive log-likelihoods on the hold-out

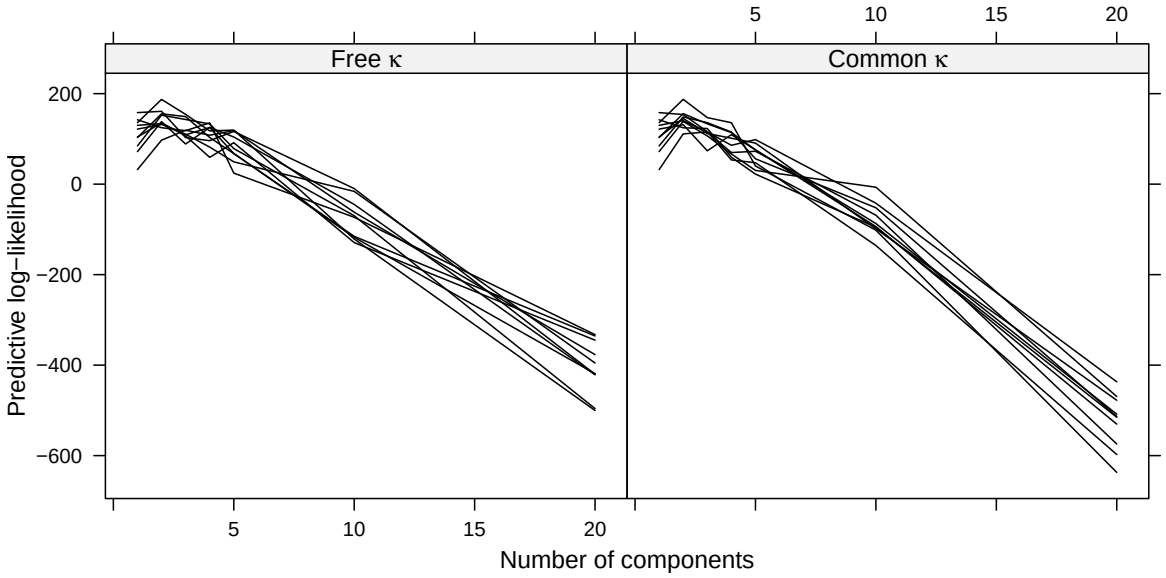

Figure 5: Predictive log-likelihoods for different and common concentration parameters κ for the fitted mixtures of vMF distributions to the "useR! 2008" abstracts.

data to select a suitable model. The numbers of components are varied and the mixtures are fitted with concentration parameters constrained to be the same over components as well as where the concentration parameters are allowed to freely vary over components. For each training data set the EM algorithm is repeated 20 times with different random initializations.

```
R> set.seed(2008)
R> library("movMF")
R> Ks <- c(1:5, 10, 20)
R> splits <- sample(rep(1:10, length.out = nrow(useR_2008_abstracts_DTM)))
R> useR_2008_movMF <- lapply(Ks, function(k)
+    sapply(1:10, function(s) {
+      m <- movMF(useR_2008_abstracts_DTM[splits != s, ], k = k, nruns = 20)
+      logLik(m, useR_2008_abstracts_DTM[splits == s, ])}))
R> useR_2008_movMF_common <- lapply(Ks, function(k)
+    sapply(1:10, function(s) {
+      m <- movMF(useR_2008_abstracts_DTM[splits != s, ],
+        k = k, nruns = 20, kappa = list(common = TRUE))
+      logLik(m, useR_2008_abstracts_DTM[splits == s, ])}))
```

In Figure 5 the fitted models are compared using the cross-validated predictive log-likelihoods. The results for the models with free concentration parameters are on the left, for the models with common concentration parameters on the right. The predictive log-likelihoods on the hold-out data indicate that the best solutions have between 1 and 5 components and that the models with more components have rather bad predictive log-likelihoods.

Following the conclusions of the comparison of the predictive log-likelihoods values we further investigate the model where the concentration parameters are constrained to be equal over components and the number of components is equal to 2.

```
R> set.seed(2008)
R> best_model <- movMF(useR_2008_abstracts_DTM, k = 2, nruns = 20,
+    kappa = list(common = TRUE))
```

In the following we look at the 10 most important words of each fitted component:

```
R> apply(coef(best_model)$theta, 1, function(x)
+    colnames(coef(best_model)$theta)[order(x, decreasing = TRUE)[1:10]])
```

```
      1          2
 [1,] "user"     "estim"
 [2,] "interfac" "variabl"
 [3,] "project"  "method"
 [4,] "gui"      "regress"
 [5,] "analysi"  "function"
 [6,] "statist"  "bayesian"
 [7,] "system"   "robust"
 [8,] "softwar"  "paramet"
 [9,] "graphic"  "test"
[10,] "report"   "spatial"
```

Clearly one component deals with issues related to infrastructure or implementation, while the other component is focusing more on statistical modeling.

The clustering obtained from the a-posteriori probabilities is analyzed by comparing the cluster membership with the keywords assigned to an abstract. Because each abstract might have several keywords assigned, the abstracts and their cluster assignments are suitably repeated.

```
R> clustering <- predict(best_model)
R> keywords <- useR_2008_abstracts[, "Keywords"]
R> keywords <- sapply(keywords, function(x)
+    sapply(strsplit(x, ", ")[[1]], function(y) strsplit(y, "-")[[1]][1]))
R> tab <- table(Keyword = unlist(keywords),
+    Component = rep(clustering, sapply(keywords, length)))
```

In the following only keywords are shown which have more than 8 abstracts assigned to them.

```
R> (tab <- tab[rowSums(tab) > 8, ])
```

```
                           Component
Keyword                      1  2
  bioinformatics             3  8
  biostatistics              1 11
  connectivity              11  0
  environmetrics             5  6
  high performance computing 13  0
  modeling                   2 12
  user interfaces           15  0
```

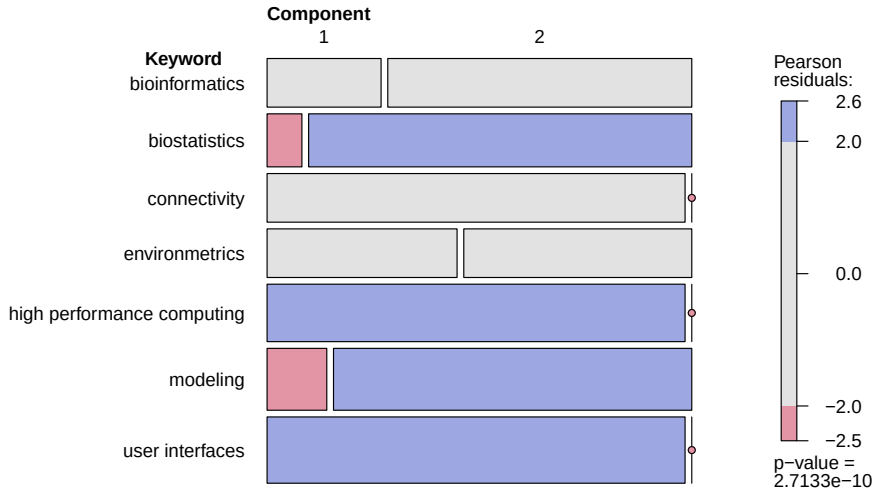

Figure 6: Cross-tabulation of the keywords in each session and the cluster assignment for keywords which were assigned to more than 8 abstracts.

The table is also visualized in Figure 6. Abstracts where the keywords assigned relate to infrastructure or implementational issues such as connectivity, high performance computing and user interfaces are associated with one component, whereas abstracts which are related to statistical modeling issues such as bioinformatics, biostatistics and modeling are more likely to be assigned to the other component.

8. Summary

An R package for fitting finite mixtures of vMF distributions is presented. Special focus has been given on numerical issues when evaluating the log-likelihood as well as on methods for ML estimation of the concentration parameter.

A possible extension for package **movMF** would be to allow for more flexible distributions in the components which also have as support the unit sphere. The vMF distribution is the analogue of the isotropic multivariate normal distribution, i.e., where the variance-covariance matrix is a multiple of the identity matrix. In $\mathbb{R}^3$ the generalization of the vMF distribution which is the analogue to the general multivariate normal distribution on the two-dimensional unit sphere is the Fisher-Bingham (or Kent) distribution (Kent 1982). Finite mixtures of Fisher-Bingham distributions were used in Peel, Whiten, and McLachlan (2001) to identify joint sets present in rock mass. They also allowed for an additional component which followed the uniform distribution on the unit sphere. However, no generalization for higher dimensions are available and Dortet-Bernadet and Wicker (2008) propose to use inverse stereographic projections of multivariate normal distributions to cluster gene expression profiles.

Acknowledgments

This research was supported by the Austrian Science Fund (FWF) under Elise-Richter grant V170-N18.

We would like to thank Achim Zeileis and Uwe Ligges for helping generating the corpus of "useR! 2008" abstracts.

References

Amos DE (1974). "Computation of Modified Bessel Functions and Their Ratios." *Mathematics of Computation*, **28**(125), 239–251.

Atkinson KE (1989). *An Introduction to Numerical Analysis*. 2nd edition. John Wiley & Sons, New York.

Banerjee A, Dhillon IS, Ghosh J, Sra S (2005). "Clustering on the Unit Hypersphere Using von Mises-Fisher Distributions." *Journal of Machine Learning Research*, **6**(September), 1345–1382.

Bangert M, Hennig P, Oelfke U (2010). "Using an Infinite von Mises-Fisher Mixture Model to Cluster Treatment Beam Directions in External Radiation Therapy." In *Proceedings of the 9th International Conference on Machine Learning and Applications (ICMLA)*, pp. 746–751.

Bates D, Maechler M (2014). ***Matrix**: Sparse and Dense Matrix Classes and Methods*. R package version 1.1-4, URL http://CRAN.R-project.org/package=Matrix.

Celeux G, Govaert G (1992). "A Classification EM Algorithm for Clustering and Two Stochastic Versions." *Computational Statistics & Data Analysis*, **14**(3), 315–332.

Cody WJ (1993). "Algorithm 715: **SPECFUN** – A Portable FORTRAN Package of Special Function Routines and Test Drivers." *ACM Transactions on Mathematical Software*, **19**(1), 22–32.

Dhillon IS, Fan J, Guan Y (2001). "Efficient Clustering of Very Large Document Collections." In RL Grossman, C Kamath, P Kegelmeyer, V Kumar, RR Namburu (eds.), *Data Mining for Scientific and Engineering Applications*, pp. 357–381. Kluwer Academic Publishers.

Dhillon IS, Modha DS (2001). "Concept Decompositions for Large Sparse Text Data Using Clustering." *Machine Learning*, **42**(1), 143–175.

Dhillon IS, Sra S (2003). "Modeling Data using Directional Distributions." *Technical Report TR-03-06*, Department of Computer Sciences, The University of Texas at Austin. URL ftp://ftp.cs.utexas.edu/pub/techreports/tr03-06.ps.gz.

Dortet-Bernadet JL, Wicker N (2008). "Model-Based Clustering on the Unit Sphere with an Illustration Using Gene Expression Profiles." *Biostatistics*, **9**(1), 66–80.

Everitt BS, Hothorn T (2014). ***HSAUR2**: A Handbook of Statistical Analyses Using R (2nd Edition)*. R package version 1.1-9, URL http://CRAN.R-project.org/package=HSAUR2.

Feinerer I, Hornik K (2014). ***tm**: Text Mining Package*. R package version 0.6, URL http://CRAN.R-project.org/package=tm.

Feinerer I, Hornik K, Meyer D (2008). "Text Mining Infrastructure in R." *Journal of Statistical Software*, **25**(5), 1–54. URL http://www.jstatsoft.org/v25/i05/.

Frühwirth-Schnatter S (2006). *Finite Mixture and Markov Switching Models*. Springer Series in Statistics. Springer-Verlag, New York.

Gautschi W, Slavik J (1978). "On the Computation of Modified Bessel Function Ratios." *Mathematics of Computation*, **32**(143), 865–875.

Hoff P (2012). ***rstiefel**: Random Orthonormal Matrix Generation on the Stiefel Manifold*. R package version 0.9, URL http://CRAN.R-project.org/package=rstiefel.

Hoff PD (2009). "Simulation of the Matrix Bingham-von Mises-Fisher Distribution, with Applications to Multivariate and Relational Data." *Journal of Computational and Graphical Statistics*, **18**(2), 438–456.

Hornik K (2005). "A CLUE for CLUster Ensembles." *Journal of Statistical Software*, **14**(12), 1–25. URL http://www.jstatsoft.org/v14/i12/.

Hornik K (2014). ***clue**: Cluster Ensembles*. R package version 0.3-48, URL http://CRAN.R-project.org/package=clue.

Hornik K, Feinerer I, Kober M (2014a). ***skmeans**: Spherical k-Means Clustering*. R package version 0.2-6, URL http://CRAN.R-project.org/package=skmeans.

Hornik K, Feinerer I, Kober M, Buchta C (2012). "Spherical k-Means Clustering." *Journal of Statistical Software*, **50**(10), 1–22. URL http://www.jstatsoft.org/v50/i10/.

Hornik K, Grün B (2013). "Amos-Type Bounds for Modified Bessel Function Ratios." *Journal of Mathematical Analysis and Applications*, **408**(1), 91–101.

Hornik K, Grün B (2014). "On Maximum Likelihood Estimation of the Concentration Parameter of von Mises-Fisher Distributions." *Computational Statistics*. doi:10.1007/s00180-013-0471-0. Forthcoming.

Hornik K, Grün B (2014). ***movMF**: Mixtures of von Mises-Fisher Distributions*. R package version 0.2-0, URL http://CRAN.R-project.org/package=movMF.

Hornik K, Meyer D, Buchta C (2014b). ***slam**: Sparse Lightweight Arrays and Matrices*. R package version 0.1-32, URL http://CRAN.R-project.org/package=slam.

Karypis G (2003). "**CLUTO**: A Clustering Toolkit." *Technical Report 02-017*, Department of Computer Science, University of Minnesota. URL http://glaros.dtc.umn.edu/gkhome/fetch/sw/cluto/manual.pdf.

Karypis G (2006). ***CLUTO** – Software for Clustering High-Dimensional Datasets*. Version 2.1.2a, URL http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview.

Kent JT (1982). "The Fisher-Bingham Distribution on the Sphere." *Journal of the Royal Statistical Society B*, **44**(1), 71–80.

Maechler M (2013). ***Bessel**: Bessel Functions Computations and Approximations*. R package version 0.5-5, URL http://CRAN.R-project.org/package=Bessel.

Mardia KV, Jupp PE (1999). *Directional Statistics*. Probability and Statistics. John Wiley & Sons.

McGraw T, Vemuri B, Yezierski R, Mareci T (2006). "Segmentation of High Angular Resolution Diffusion MRI Modeled as a Field of von Mises-Fisher Mixtures." In A Leonardis, H Bischof, A Pinz (eds.), *Computer Vision – ECCV 2006*, volume 3953 of *Lecture Notes in Computer Science*, pp. 463–475. Springer-Verlag, Berlin.

McLachlan GJ, Peel D (2000). *Finite Mixture Models*. John Wiley & Sons.

Olver FWJ (1954). "The Asymptotic Expansion of Bessel Functions of Large Order." *Philosophical Transactions of the Royal Society of London A*, **247**(930), 328–368.

Peel D, Whiten WJ, McLachlan GJ (2001). "Fitting Mixtures of Kent Distributions to Aid in Joint Set Identification." *Journal of the American Statistical Association*, **96**(453), 56–63.

Press WH, Teukolsky SA, Vetterling WT, Flannery BP (2002). *Numerical Recipes in C: The Art of Scientific Computing*. 2nd edition. Cambridge University Press.

R Core Team (2014). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

Salton G, Buckley C (1988). "Term-Weighting Approaches in Automatic Text Retrieval." *Information Processing and Management*, **24**(5), 513–523.

Schou G (1978). "Estimation of the Concentration Parameter in von Mises-Fisher Distributions." *Biometrika*, **65**(1), 369–377.

Simpson HC, Spector SJ (1984). "Some Monotonicity Results for Ratios of Modified Bessel Functions." *Quarterly of Applied Mathematics*, **42**(1), 95–98.

Song H, Liu J, Wang G (2012). "High-Order Parameter Approximation for von Mises-Fisher Distributions." *Applied Mathematics and Computation*, **218**(24), 11880–11890.

Sra S (2012). "A Short Note on Parameter Approximation for von Mises-Fisher Distributions: and a Fast Implementation of $I_s(x)$." *Computational Statistics*, **27**(1), 177–190.

Strehl A, Ghosh J (2002). "Cluster Ensembles – A Knowledge Reuse Framework for Combining Multiple Partitions." *Journal of Machine Learning Research*, **3**(December), 583–617.

Tanabe A, Fukumizu K, Oba S, Takenouchi T, Ishii S (2007). "Parameter Estimation for von Mises-Fisher Distributions." *Computational Statistics*, **22**(1), 145–157.

Tang H, Chu SM, Huang TS (2009). "Generative Model-Based Speaker Clustering via Mixture of von Mises-Fisher Distributions." In *Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 4101–4104. IEEE Computer Society, Washington, DC, USA.

Tretter MJ, Walster GW (1980). "Further Comments on the Computation of Modified Bessel Function Ratios." *Mathematics of Computation*, **35**(151), 937–939.

Ulrich G (1984). "Generation of Distributions on the m-Sphere." *Journal of the Royal Statistical Society C*, **33**(2), 158–163.

Watson GN (1995). *A Treatise on the Theory of Bessel Functions*. Cambridge Mathematical Library, 2nd edition. Cambridge University Press.

Wood ATA (1994). "Simulation of the von Mises Fisher Distribution." *Communications in Statistics – Simulation and Computation*, **23**(1), 157–164.

Zhao Y, Karypis G (2001). "Criterion Functions for Document Clustering: Experiments and Analysis." *Technical Report TR #01-40*, Department of Computer Science, University of Minnesota.

Zhao Y, Karypis G (2004). "Empirical and Theoretical Comparisons of Selected Criterion Functions for Document Clustering." *Machine Learning*, **55**(3), 311–331.

Affiliation:

Kurt Hornik
Institute for Statistics and Mathematics
WU Wirtschaftsuniversität Wien
Welthandelsplatz 1
1020 Wien, Austria
E-mail: Kurt.Hornik@R-project.org
URL: http://statmath.wu.ac.at/~hornik/

Bettina Grün
Institut für Angewandte Statistik / IFAS
Johannes Kepler Universität Linz
Altenbergerstraße 69
4040 Linz, Austria
E-mail: Bettina.Gruen@jku.at
URL: http://ifas.jku.at/gruen/

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume 58, Issue 10 *Submitted:* 2012-10-23
July 2014 *Accepted:* 2014-03-20