A Quick Guide for the EMCluster Package

Wei-Chen Chen[1], **Volodymyr Melnykov**[2], **Ranjan Maitra**[3]

[1]Computer Science and Mathematics Division,
Oak Ridge National Laboratory,
Oak Ridge, TN, USA

[2]Department of Information Systems, Statistics, and Management Science,
The University of Alabama,
Tuscaloosa, AL, USA

[3]Department of Statistics,
Iowa State University,
Ames, IA, USA

Contents

Acknowledgement

The authors acknowledge partial support by the National Science Foundation CAREER Grant # DMS-0437555.

Warning: This document is written to explain the main functions of **EMCluster** (Chen *et al.* 2012), version 0.2-0. Every effort will be made to ensure future versions are consistent with these instructions, but features in later versions may not be explained in this document.

1. Introduction

We use this section to show how to install **EMCluster**, introduce basic finite mixture Gaussian distribution, and notation for the corresponding R objects. In Section 2, we introduce EM algorithms and strategies of initialization for model-based clustering, and the major R functions are also introduced. In Section 3, we provide two examples for unsupervised and semi-supervised clusterings, and quick demos are shown.

1.1. Installation

The **EMCluster** has simple interface of R to efficient C code that we optimize the algorithm and utilize **LAPACK** for matrix algebra. The package should install on most popular platform with further configureations. The installation can be done in the shell command as

Shell Command
```
> R CMD INSTALL EMCluster_0.2-0.tar.gz
```

or from any R session as

Shell Command
```
R> install.packages("EMCluster")
```

with user-favored CRAN mirror site.

1.2. Notation

The **EMCluster** assumes finite mixture Gaussian distribution with unstructured dispersion and implements EM algorithm for model-based clustering in both unsupervised and semi-supervised clusterings. The model is

$$f(\boldsymbol{x}|\boldsymbol{\vartheta}) = \sum_{k=1}^{K} \pi_k \phi(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k). \tag{1}$$

- $\boldsymbol{x}$ is a p-dimensional observation.
- $\boldsymbol{\vartheta} = \{\pi_1, \pi_2, \ldots, \pi_{K-1}, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_K, \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2, \ldots, \boldsymbol{\Sigma}_K\}$.
- $\pi_1, \pi_2, \ldots, \pi_K$ are mixing proportion for K components that $\sum_{k=1}^K \pi_k = 1$ and $0 < \pi_k < 1$ for all $k = 1, 2, \ldots, K$.
- $\phi(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$'s are multivariate Gaussian densities with mean $\boldsymbol{\mu}_k$ and dispersion $\boldsymbol{\Sigma}_k$.
- $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_K$ are p-dimensional mean vectors and $\boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2, \ldots, \boldsymbol{\Sigma}_K$ are $p \times p$-dimensional dispersion matrices.

We also assume the following notation for R objects in **EMCluster**

- `x` is a matrix with dimension $n \times p$ containing n observations to be clustered.
- `pi` is a vector with length K containing $\pi_1, \pi_2, \dots, \pi_K$.
- `Mu` is a matrix with dimension $K \times p$ containing $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_K$.
- `LTSigma` is a matrix with dimension $K \times p(p+1)/2$ containing low triangular matrices of $\boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2, \dots, \boldsymbol{\Sigma}_K$.
- `lab` is a vector with length n containing labels of observations. If `lab=NULL`, then unsupervised clustering is performed, otherwise semi-supervised clustering is performed where labels can be $1, 2, \dots, K$ for cluster-known data and 0 for cluster-unknown data.

2. EM Algorithm and Initialization

For n observations $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n\}$, the log likelihood based on the Equation (1) is

$$\log L(\boldsymbol{\vartheta}|\boldsymbol{X}) = \log \left(\prod_{i=1}^{n} f(\boldsymbol{x}_i|\boldsymbol{\vartheta}) \right). \tag{2}$$

A standard way to optimize the Equation (2) is to utilize the EM algorithm (Dempster *et al.* 1977) composed of expectation (E-) and maximization (M-) steps.

The R function `emcluster` implements the algorithm to find the maximum likelihood estimates (MLEs) $\hat{\boldsymbol{\vartheta}}$. The R functions `e.step` and `m.step` implements the both steps, respectively, which are useful for advanced developers.

Model-based clustering partition data into K clusters by the maximum posterior

$$\operatorname*{argmax}_k \hat{\pi}_k \phi(\boldsymbol{x}_i|\hat{\boldsymbol{\mu}}_k, \hat{\boldsymbol{\Sigma}}_k)$$

for each observation. The R function `assign.class` provides this procedure returning cluster id $(1, 2, \dots, K)$ for each $\boldsymbol{x}_i$.

The initialization is an solution of avoiding local optimization when applying the EM algorithm to cluster high-dimensional data. The local optimization problem is more servious when data contain high overlaps of clusters. There are three initialization methods implemented in **EMCluster**: *RndEM* (Maitra 2009), *emEM* (Biernacki *et al.* 2003), and *svd* (Maitra 2001). The R functions are

- `init.EM` calls `rand.EM` and `em.EM` and implements for *RndEM* and *emEM* methods, respectively, and
- `emgroup` implements the *svd* method.

The short summary of the initialization methods is

- *RndEM* repeats `.EMC$short.iter` times of randomly selecting K centers from data and group all other data to the closest center based on Euclidean distance. Pick the best initial in terms of the highest log likelihood. Then, start EM algorithm from the best initial until convergence.
- *emEM* is composed of short-EM and long-EM steps. The short-EM step starts with randomly selecting K centers and applies EM iterations until convergence with loose tolerance `.EMC$short.eps`. Repeat a few time until the total EM iterations reach `.EMC$short.iter`. Then, start long-EM algorithm from the best initial until convergence with tight tolerance `.EMC$EM.eps`.
- *svd* utilizes singular value decomposition to decomposition data, and select centers from the major component space determining by the singular values. The data are grouped to the centers by kmeans to generate an initial. Then, start EM algorithm from the initial until convergence.

3. Examples

3.1. EM Control

The **EMCluster** provides a control function `.EMControl` and three default controls `.EMC`, `.EMC.Rnd`, and `.EMC.Rndp` which are useful to manipulate different initializations. By default `.EMC` is for *emEM*, `.EMC.Rnd` is for *RndEM*, while `.EMC.Rndp` is for *RndEM* but with different flavor.

For larger data or high overlaps data, general users may want to increase the initial iteration (`.EMC$short.iter`) which can improve the clustering results. However, the initialization also cost computing time and may take longer to obtain stable initials especially for large number of clusters or high overlap data.

Anyway, these also provide advanced developers to explore new initialization methods. We give two examples next for unsupervised and semi-supervised clusterings utilizing all initialization methods in **EMCluster**.

3.2. Unsupervised Clustering

In an R session, you can run the demo as

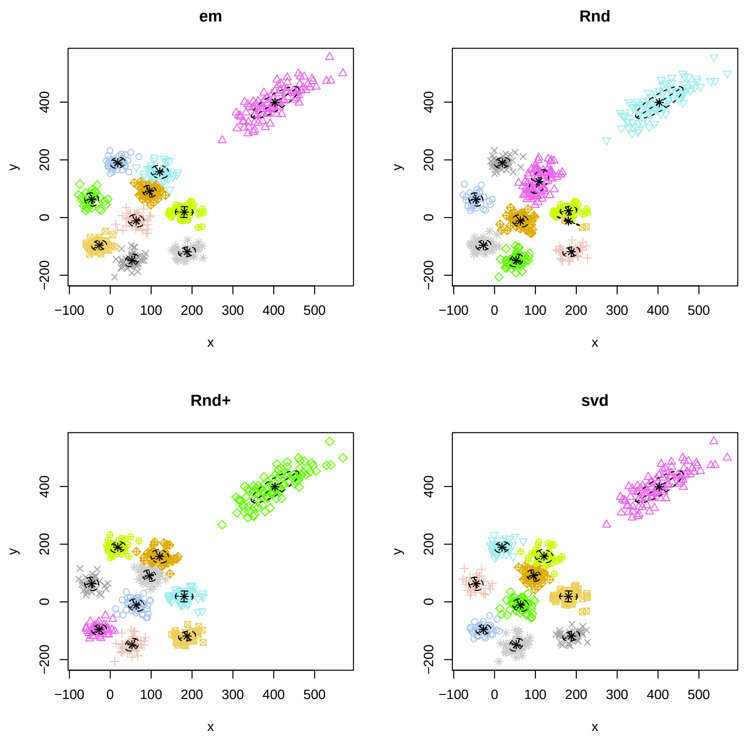
unsupervised

```
R> demo(allinit, 'EMCluster', ask = F)
```

which clusters a small dataset `da1` of **EMCluster** with 10 clusters. The data set has only 500 observations in two dimensions.

This demo provides a plot with four unsupervised clustering results including *emEM* (em), *RndEM* (Rnd), *RndEM+* (Rnd+) and *svd* (svd) where colors and symbols indicate different clusters as in the Figure 1. The *RndEM+* has different settings to the *RndEM*. The demo also compares classifications with true ids using adjusted Rand index (Hubert and Arabie 1985) which has value before 0 (bad results) and 1 (perfect match) as in the Table 1.

Figure 1: Unsupervised clustering results for four initializations.



	logL	adjR
em	-5657.603	0.9929422
Rnd	-5658.984	0.9096019
Rnd+	-5657.612	0.9929422
svd	-5657.600	0.9929422

Table 1: Comparison by log likelihood (logL) and adjusted Rand index (adjR) of four initializations.

3.3. Semi-supervised Clustering

Inside an R session, you can run the demo as

unsupervised

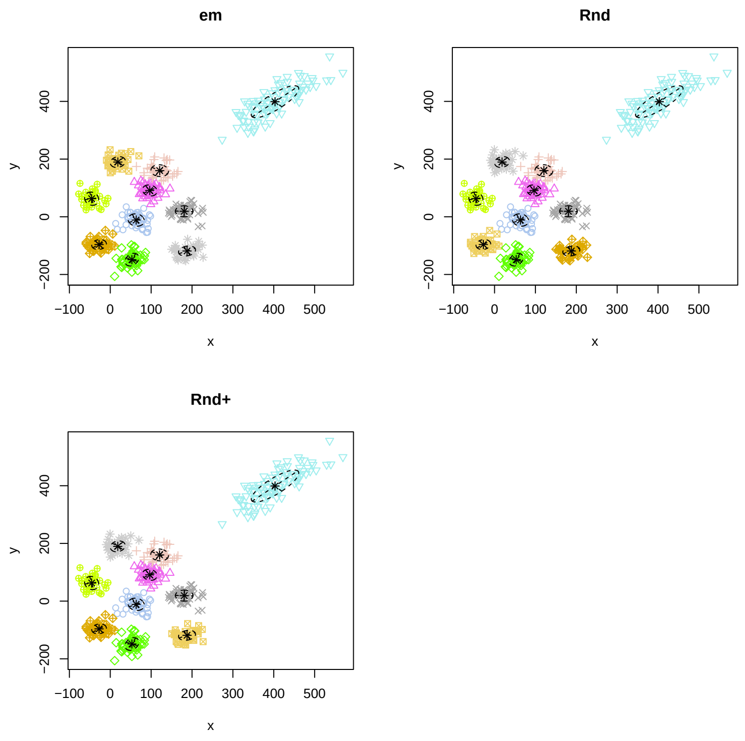
```
R> demo(allinit_ss, 'EMCluster', ask = F)
```

which clusters a small dataset `da1` in **EMCluster** with 10 clusters.

We randomly choose 6 clusters, then select 50% of data for each cluster. There are about 170 points are labeled with true which are prior information, then we cluster the rest 330 points into 10 clusters by semi-supervised clustering.

This demo provides a plot with three semi-supervised clustering results including *emEM* (em), *RndEM* (Rnd), *RndEM+* (Rnd+) where colors and symbols indicate different clusters as in the Figure 2. Note that *svd* method is not implemented for semi-supervised clustering. The demo also compares classifications with true ids using adjusted Rand index as in the Table 2. Comparing with the Table 1, the semi-supervised clustering provides better results than the unsupervised clustering in terms of higher log likelihood (logL) and adjusted Rand index (adjR). The reason is that some tiny clusters are not seeded at the initial step in unsupervised clustering, therefore, a few prior information from there can substantially improve accuracy by the semi-supervised clustering.

Figure 2: Semi-supervised clustering results for three initializations.



	logL	adjR
em	-5657.882	0.9938959
Rnd	-5657.880	0.9938959
Rnd+	-5657.881	0.9938959

Table 2: Comparison by log likelihood (logL) and adjusted Rand index (adjR) of three initializations.

References

Biernacki C, Celeux G, Govaert G (2003). "Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate Gaussian mixture models." *Computational Statistics and Data Analysis*, **413**, 561–575.

Chen WC, V M, Maitra R (2012). "EMCluster: EM Algorithm for Model-Based Clustering of Finite Mixture Gaussian Distribution." R Package, URL http://cran.r-project.org/package=EMCluster.

Dempster A, Laird N, Rubin D (1977). "Maximum Likelihood Estimation from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society Series B*, **39**(3), 1–38.

Hubert L, Arabie P (1985). "Comparing partitions." *Journal of Classification*, **2**, 193–218.

Maitra R (2001). "Clustering massive datasets with applications to software metrics and tomography." *Technometrics*, **43**(3), 336–346.

Maitra R (2009). "Initializing Partition-Optimization Algorithms." *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, **6**, 144–157. doi:http://doi.ieeecomputersociety.org/10.1109/TCBB.2007.70244.