mixComp: An R package for estimating complexity of a mixture

Anja Weigel²*, Fadoua Balabdaoui ¹*, Yulia Kulagina ¹*, and Lilian Mueller²*

¹ ETH Zurich, Seminar for Statistics, Switzerland ² ETH Zurich, Switzerland ¶ Corresponding author
* These authors contributed equally.

DOI: 10.21105/joss.04354

Software
- Review
- Repository
- Archive

Summary
Mixture models (see Lindsay, 1983a, 1983b; McLachlan & Peel, 2000; Teicher, 1963; Titterington et al., 1985) allow for modeling heterogeneous data. The number of mixture components may be known in advance, in which case the model parameters can be easily estimated (e.g., their maximum likelihood estimates (MLE) can be computed using the EM (Expectation-Maximization) algorithm (Dempster et al., 1977)). However, in many applications the number of components is unknown and has to be inferred from the data.

mixComp provides three categories of methods for estimating the unknown complexity of a (univariate) finite mixture:

- methods built upon the determinants of the Hankel matrix of moments of the mixing distribution;
- methods based on penalized minimum distance between the unknown probability density and its consistent estimator;
- likelihood ratio test (LRT) - based techniques.

All methods come with theoretical guarantees for consistency.

Statement of need
mixComp is aimed at practitioners studying phenomena that can be effectively modelled using mixture distributions. Two main features distinguish it from other mixture-related R (R Core Team, 2020) packages:

- while mixture component weights and parameters are often estimated as a by-product, mixComp methods are based on theory specifically developed to consistently estimate mixture complexity;
- mixComp is applicable to parametric mixtures beyond those whose component distributions are included in the stats package, making it more customizable than most packages for model-based clustering.

Other packages dealing with mixture models are mclust (Scrucca et al., 2016), which fits Gaussian mixtures using the EM algorithm, MixSim (Melnikov et al., 2012), which allows for simulating from mixtures and comparing the performance of clustering algorithms, and mixdist (Macdonald & Du, 2018), used for grouped conditional data. mixtools (Benaglia et al., 2009) focuses on mixture-of-regressions and non-parametric mixtures and is used to fit (multivariate) normal, multinomial or gamma mixtures with the EM algorithm, also containing routines for selecting the number of components based on information criteria and parametric
https://doi.org/10.21105/joss.04354.

bootstrapping of the LRT statistic values. However, they are limited to multinomial, normal mixtures and mixtures-of-regressions. *flexmix* (Grün & Leisch, 2007, 2008; Leisch, 2004) handles mixtures-of-regression and stands out due to its extensibility, a design principle that we also aimed for. *rebmix*(Nagode, 2018), dealing with univariate and multivariate finite mixture model for generation, estimation, clustering, classification purposes, offers a wide variety of recognized methods for mixture model estimation for both discrete and continuous variables. The approaches suggested in mixComp are however not among those used in *rebmix*, thus complementing it rather than providing competition.

### Methods

A distribution $F$ is called a *finite mixture* if its probability density/mass is of the form

$$f(x) = \sum_{i=1}^{p} w_i g_i(x, \theta_i),$$

$p \in \mathbb{N}$ being the mixture complexity, $(w_1, \ldots, w_p : \sum_{i=1}^{p} w_i = 1, w_i \geq 0, \text{ for } i = 1, \ldots, p)$ - component weights and $g_i(x, \theta_i)$ - $i$-th component density.

Given some complexity $j$, the relevant parameter spaces are

$$\Theta_j = \{\theta_1, \ldots, \theta_j : \theta_i \in \Theta \subseteq \mathbb{R}^d, \quad d \in \mathbb{N}, \quad \text{for } i = 1, \ldots, j\},$$

$$W_j = \{w_1, \ldots, w_j : \sum_{i=1}^{j} w_i = 1, w_i \geq 0, \text{ for } i = 1, \ldots, j\}.$$

Assume the family of the component densities $\{g(x; \theta)\}$ is known, $\theta = (\theta_1, \ldots, \theta_p) \in \Theta_p$, $w = (w_1, \ldots, w_p) \in W_p$ and $p \in \mathbb{N}$ are unknown.

1. **Functions using Hankel matrices**

The basic Hankel approach (Dacunha-Castelle & Gassiat, 1997) estimates (based on $X = \{X_1, \ldots, X_n\}$, an i.i.d. $n$-sample from $F$)

$$\hat{\theta} := \arg\min_{\theta \in \Theta} \left\{ |\det H(\tilde{c}_{2j+1})| + A(j)l(n) \right\},$$

with positive function $l(n) \to 0$ as $n \to \infty$; positive, strictly increasing function $A(j)$; $H(\tilde{c}_{2j+1})$ - Hankel matrix built on $\tilde{c}_{2j+1}$, the consistent estimator of the first $2j+1$ moments of the mixing distribution.

**mixComp** offers several methods for calculating $\tilde{c}_{2j+1}$ and provides extensions of the basic approach.

2. **Functions using distances**

Consider the parametric family

$$\mathcal{F}_j = \{f_{j, \omega, \theta} : (\omega, \theta) \in W_j \times \Theta_j\},$$

$$f_{j, \omega, \theta}(x) = \sum_{i=1}^{j} w_i g(x; \theta_i), \quad \{g(x; \theta) : \theta \in \Theta\}. \quad \text{Note: } \mathcal{F}_j \subseteq \mathcal{F}_{j+1}, \forall j = 1, 2, \ldots.$$  

These methods search for the ‘best’ estimate (e.g. MLE) $(\hat{\omega}, \hat{\theta}) \in W_j \times \Theta_j$ for a given $j$ and thereby specified density/mass function $f_j(x) = f_{j, \hat{\omega}, \hat{\theta}}(x)$, and the non-parametric density/mass estimate $\hat{f}_n(x)$. Then

$$\hat{\rho} = \min_j \left\{ D(f_j, \hat{f}_n) - D(\hat{f}_{j+1}, \hat{f}_n) \leq l(j, n) \right\},$$
where $D$ denotes the distance measure, $t(j,n)$ - a suitable penalty function.

**mixComp** offers several distance-based procedures following (Umashanger & Sriram, 2009; Woo & Sriram, 2006; Woo & Sriram, 2007).

### 3. Functions using LRTS

These methods obtain the MLE for the mixture density/mass with $j$ and $j+1$ components ($j = 1, 2, \ldots$), yielding $(\hat{w}_j, \hat{\theta}_j) \in W_j \times \Theta_j$ and $(\hat{w}_{j+1}, \hat{\theta}_{j+1}) \in W_{j+1} \times \Theta_{j+1},$

\[
\text{LRTS} = -2 \ln \left( \frac{L_X(\hat{w}_j, \hat{\theta}_j)}{L_X(\hat{w}_{j+1}, \hat{\theta}_{j+1})} \right),
\]

$L_X$ being the likelihood function given $X$.

A parametric bootstrap sampling is applied to generate $B$ $n$-samples from a $j$-component mixture given $(\hat{w}_j, \hat{\theta}_j)$. For each bootstrap sample, the method computes the MLEs and the LRTS (Likelihood Ratio Test Statistic) corresponding to the mixture densities with $j$ and $j+1$ components. The decision is to reject $H_0: p = j$, setting $j \leftarrow j + 1$ if the LRTS is larger than the specified quantile of its bootstrapped counterparts; otherwise $\hat{p}$ is set to $j$ (Xekalaki & Karlis, 1999).

### Examples

Hellinger distance method with bootstrap applied to the Shakespeare data (viewed as a mixture of geometrics) (Balabdaoui & Kulagina, 2020; Chee & Wang, 2016; Efron & Thisted, 1976; Spevack, 1968).

```r
# apply the shift:
shakespeare.obs <- unlist(shakespeare) - 1
# define the MLE function:
MLE.geom <- function(dat) 1 / (mean(dat) + 1)
# create the datMix object:
Shakespeare.dM <- datMix(shakespeare.obs, dist = "geom", discrete = TRUE,
                         MLE.function = MLE.geom, theta.bound.list = list(prob = c(0, 1)))
# estimate the complexity:
set.seed(0)
(res <- hellinger.boot.disc(Shakespeare.dM, B = 50, ql = 0.025, qu = 0.975))
> The estimated order is 3.
# plot:
plot(res, breaks = 100, xlim = c(0, 20))
```

---

Estimated 3 component 'geom' mixture model

Figure 1: Hellinger distance method with bootstrap for the Shakespeare data

LRT method applied to the Acidity data (Crawford et al., 1992; Crawford, 1994; Richardson & Green, 1997).

```r
# define the MLE functions:
MLE.norm.mean <- function(dat) mean(dat)
MLE.norm.sd <- function(dat){
  sqrt((length(dat) - 1) / length(dat)) * sd(dat)
}
MLE.norm.list <- list("MLE.norm.mean" = MLE.norm.mean,
                       "MLE.norm.sd" = MLE.norm.sd)
# define parameter bounds:
norm.bound.list <- list("mean" = c(-Inf, Inf), "sd" = c(0, Inf))
acidity.obs <- unlist(acidity)
# create the datMix object:
acidity.dM <- datMix(acidity.obs, dist = "norm", discrete = FALSE,
                      MLE.function = MLE.norm.list, theta.bound.list = norm.bound.list)
# estimate the complexity:
set.seed(0)
res <- mix.lrt(acidity.dM, B = 100, quantile = 0.95)
```

Estimated 2 component 'norm' mixture model

Figure 2: LRT method for the Acidity data

References


Spevack, M. (1968). *A complete and systematic concordance to the works of shakespeare*.


