Package ‘bmixture’

April 23, 2019

Version 1.3
Date 2019-04-23
Title Bayesian Estimation for Finite Mixture of Distributions
Author Reza Mohammadi <https://orcid.org/0000-0001-9538-0648>
Maintainer Reza Mohammadi <a.mohammadi@uva.nl>

URL https://www.uva.nl/profile/a.mohammadi

Description Provides statistical tools for Bayesian estimation for finite mixture of distributions, mainly mixture of Gamma, Normal and t-distributions. The package is implemented the recent improvements in Bayesian literature for the finite mixture of distributions, including Mohammadi and et al. (2013) <doi:10.1007/s00180-012-0323-3> and Mohammadi and Salehi-Rad (2012) <doi:10.1080/03610918.2011.588358>.

Depends R (>= 3.0.0)
Imports BDgraph
License GPL (>= 2)
Repository CRAN
NeedsCompilation yes
Date/Publication 2019-04-23 16:40:03 UTC

R topics documented:

bmixture-package .................................................. 2
bmixgamma ......................................................... 3
bmixnorm .......................................................... 6
bmixt ............................................................... 9
galaxy .............................................................. 11
mixgamma .......................................................... 12
mixnorm ........................................................... 14
mixt ............................................................... 15
plot.bmixgamma .................................................. 17
plot.bmixnorm ................................................... 18
plot.bmixt ........................................................ 19
Description

The R package bmixture provides statistical tools for Bayesian estimation in finite mixture of distributions. The package implemented the improvements in the Bayesian literature, including Mohammadi and Salehi-Rad (2012) and Mohammadi et al. (2013). Besides, the package contains several functions for simulation and visualization, as well as a real dataset taken from the literature.

How to cite this package

Whenever using this package, please cite as


Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

References


Examples

```r
## not run:
library(bmixture)
data(galaxy)

# Running bdmcmc algorithm for the galaxy dataset
mcmc_sample = bmixnorm(data = galaxy)

summary(mcmc_sample)
plot(mcmc_sample)
print(mcmc_sample)

# Simulating data from mixture of Normal with 3 components
n = 500
mean = c(0, 10, 3)
sd = c(1, 1, 1)
weight = c(0.3, 0.5, 0.2)
data = rmixnorm(n = n, weight = weight, mean = mean, sd = sd)

# plot for simulation data
hist(data, prob = TRUE, nclass = 30, col = "gray")
x = seq(-20, 20, 0.05)
densmixnorm = dmixnorm(x, weight, mean, sd)
lines(x, densmixnorm, lwd = 2)

# Running bdmcmc algorithm for the above simulation data set
bmixnorm.obj = bmixnorm(data = data, k = 3, iter = 1000)

summary(bmixnorm.obj)
## End(not run)
```

---

**bmixgamma**

**Sampling algorithm for mixture of distributions**

**Description**

This function consists of several sampling algorithms for Bayesian estimation for finite mixture of Gamma distributions.
Usage

```r
bmixgamma( data, k = "unknown", iter = 1000, burnin = iter / 2, lambda = 1,
        mu = NULL, nu = NULL, kesi = NULL, tau = NULL, k.start = NULL,
        alpha.start = NULL, beta.start = NULL, pi.start = NULL,
        k_max = 30, trace = TRUE )
```

Arguments

data: The vector of data with size n.
k: The number of components of mixture distribution. Default is "unknown". It can take an integer values.
iter: The number of iteration for the sampling algorithm.
burnin: The number of burn-in iteration for the sampling algorithm.
lambda: For the case k = "unknown", it is the parameter of the prior distribution of number of components k.
mu: The parameter of alpha in mixture distribution.
nu: The parameter of alpha in mixture distribution.
kesi: The parameter of beta in mixture distribution.
tau: The parameter of beta in mixture distribution.
k.start: For the case k = "unknown", initial value for number of components of mixture distribution.
alpha.start: Initial value for parameter of mixture distribution.
beta.start: Initial value for parameter of mixture distribution.
pi.start: Initial value for parameter of mixture distribution.
k_max: For the case k = "unknown", maximum value for the number of components of mixture distribution.
trace: Logical: if TRUE (default), tracing information is printed.

Details

Sampling from finite mixture of Gamma distribution, with density:

\[ Pr(x | k, \pi, \alpha, \beta) = \sum_{i=1}^{k} \pi_i Gamma(x | \alpha_i, \beta_i), \]

where \( k \) is the number of components of mixture distribution (as a default we assume is unknown) and

\[ Gamma(x | \alpha_i, \beta_i) = \frac{\beta_i^{\alpha_i}}{\Gamma(\alpha_i)} x^{\alpha_i - 1} e^{-\beta_i x}. \]

The prior distributions are defined as below

\[ P(K = k) \propto \frac{\lambda^k}{k!}, \quad k = 1, ..., k_{max}, \]

\[ \pi_i | k \sim Dirichlet(1, ..., 1), \]
\[ \alpha_i | k \sim \text{Gamma}(\nu, \upsilon), \]
\[ \beta_i | k \sim \text{G}(\eta, \tau), \]

for more details see Mohammadi et al. (2013).

**Value**

An object with S3 class "bmixgamma" is returned:

- **all_k**: A vector which includes the waiting times for all iterations. It is needed for monitoring the convergence of the BD-MCMC algorithm.
- **all_weights**: A vector which includes the waiting times for all iterations. It is needed for monitoring the convergence of the BD-MCMC algorithm.
- **pi_sample**: A vector which includes the MCMC samples after burn-in from parameter \( \pi \) of mixture distribution.
- **alpha_sample**: A vector which includes the MCMC samples after burn-in from parameter \( \alpha \) of mixture distribution.
- **beta_sample**: A vector which includes the MCMC samples after burn-in from parameter \( \beta \) of mixture distribution.
- **data**: original data.

**Author(s)**

Reza Mohammadi <a.mohammadi@uva.nl>

**References**


**See Also**

`bmixnorm`, `bmixt`, `bmixgamma`
Examples

```r
## Not run:
# simulating data from mixture of gamma with two components
n = 1000 # number of observations
weight = c( 0.6, 0.4 )
alpha = c( 12, 1 )
beta = c( 3, 2 )
data <- rmixgamma( n = n, weight = weight, alpha = alpha, beta = beta )

# plot for simulation data
hist( data, prob = TRUE, nclass = 50, col = "gray" )
x = seq( 0, 10, 0.05 )
truth = dmixgamma( x, weight, alpha, beta )
lines( x, truth, lwd = 2 )

# Running bdmcmc algorithm for the above simulation data set
bmixgamma.obj <- bmixgamma( data, iter = 1000 )
summary( bmixgamma.obj )
plot( bmixgamma.obj )

## End(Not run)
```

---

**bmixnorm**

*Sampling algorithm for mixture of distributions*

**Description**

This function consists of several sampling algorithms for Bayesian estimation for finite mixture of Normal distributions.

**Usage**

```r
bmixnorm( data, k = "unknown", iter = 1000, burnin = iter / 2, lambda = 1,
k.start = NULL, mu.start = NULL, sig.start = NULL, pi.start = NULL, k_max = 30, trace = TRUE )
```

**Arguments**

- **data**
  The vector of data with size n.
- **k**
  The number of components of mixture distribution. Default is "unknown". It can take an integer values.
- **iter**
  The number of iteration for the sampling algorithm.
- **burnin**
  The number of burn-in iteration for the sampling algorithm.
lambda

For the case \( k = \text{"unknown"} \), it is the parameter of the prior distribution of number of components \( k \).

\( k \text{.start} \)

For the case \( k = \text{"unknown"} \), initial value for number of components of mixture distribution.

\( \text{mu.start} \)

Initial value for parameter of mixture distribution.

\( \text{sig.start} \)

Initial value for parameter of mixture distribution.

\( \text{pi.start} \)

Initial value for parameter of mixture distribution.

\( \text{k_max} \)

For the case \( k = \text{"unknown"} \), maximum value for the number of components of mixture distribution.

\( \text{trace} \)

Logical: if TRUE (default), tracing information is printed.

Details

Sampling from finite mixture of Gamma distribution, with density:

\[
Pr(x|k, \pi, \mu, \sigma) = \sum_{i=1}^{k} \pi_i N(x|\mu_i, \sigma_i),
\]

where \( k \) is the number of components of mixture distribution (as a default we assume is unknown). The prior distributions are defined as below

\[
P(K = k) \propto \frac{\lambda^k}{k!}, \quad k = 1, ..., k_{max},
\]

\[
\pi_i|k \sim \text{Dirichlet}(1, ..., 1),
\]

\[
\alpha_i|k \sim \text{Gamma}(\nu, \upsilon),
\]

\[
\beta_i|k \sim \text{G}(\eta, \tau),
\]

for more details see Mohammadi et al. (2013) and Mohammadi and Salehi-Rad (2012).

Value

An object with S3 class "bmixnorm" is returned:

\( \text{all_k} \)

A vector which includes the waiting times for all iterations. It is needed for monitoring the convergence of the BD-MCMC algorithm.

\( \text{all_weights} \)

A vector which includes the waiting times for all iterations. It is needed for monitoring the convergence of the BD-MCMC algorithm.

\( \text{pi_sample} \)

A vector which includes the MCMC samples after burn-in from parameter \( \pi \) of mixture distribution.

\( \text{mu_sample} \)

A vector which includes the MCMC samples after burn-in from parameter \( \mu \) of mixture distribution.

\( \text{sig_sample} \)

A vector which includes the MCMC samples after burn-in from parameter \( \sigma \) of mixture distribution.

\( \text{data} \)

The original data.
Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

References


See Also

`bmixt`, `bmixgamma`, `rmixnorm`

Examples

```r
## Not run:
data( galaxy )

# Running bdmcmc algorithm for the galaxy dataset
mcmc_sample = bmixnorm( data = galaxy )

summary( mcmc_sample )
plot( mcmc_sample )
print( mcmc_sample )

# simulating data from mixture of Normal with 3 components
n = 500
weight = c( 0.3, 0.5, 0.2 )
mean = c( 0, 10, 3 )
sd = c( 1, 1, 1 )
data = rmixnorm( n = n, weight = weight, mean = mean, sd = sd )

# plot for simulation data
hist( data, prob = TRUE, nclass = 30, col = "gray" )
```
```r
x = seq(-20, 20, 0.05)
densmixnorm = dmixnorm(x, weight, mean, sd)
lines(x, densmixnorm, lwd = 2)

# Running bdmcmc algorithm for the above simulation data set
bmixnorm.obj = bmixnorm(data, k = 3, iter = 1000)
summary(bmixnorm.obj)

## End(Not run)
```

---

**bmixt**

*Sampling algorithm for mixture of distributions*

**Description**

This function consists of several sampling algorithms for Bayesian estimation for finite mixture of Normal distributions.

**Usage**

```r
bmixt(data, k = "unknown", iter = 1000, burnin = iter / 2, lambda = 1, df = 1,
k.start = NULL, mu.start = NULL, sig.start = NULL, pi.start = NULL,
k_max = 30, trace = TRUE)
```

**Arguments**

- `data` The vector of data with size n.
- `k` The number of components of mixture distribution. Default is "unknown". It can take an integer values.
- `iter` The number of iteration for the sampling algorithm.
- `burnin` The number of burn-in iteration for the sampling algorithm.
- `lambda` For the case k = "unknown", it is the parameter of the prior distribution of number of components k.
- `df` Degrees of freedom (> 0, maybe non-integer). df = Inf is allowed.
- `k.start` For the case k = "unknown", initial value for number of components of mixture distribution.
- `mu.start` Initial value for parameter of mixture distribution.
- `sig.start` Initial value for parameter of mixture distribution.
- `pi.start` Initial value for parameter of mixture distribution.
- `k_max` For the case k = "unknown", maximum value for the number of components of mixture distribution.
- `trace` Logical: if TRUE (default), tracing information is printed.
Details

Sampling from finite mixture of Gamma distribution, with density:

\[
Pr(x|k, \pi, \mu, \sigma) = \sum_{i=1}^{k} \pi_i N(x|\mu_i, \sigma_i),
\]

where \(k\) is the number of components of mixture distribution (as a default we assume is unknown). The prior distributions are defined as below:

\[
P(K = k) \propto \frac{\lambda^k}{k!}, \quad k = 1, ..., k_{\text{max}},
\]

\[
\pi_i | k \sim \text{Dirichlet}(1, ..., 1),
\]

\[
\alpha_i | k \sim \text{Gamma}(\nu, \nu),
\]

\[
\beta_i | k \sim G(\eta, \tau),
\]

for more details see Mohammadi et al. (2013) and Mohammadi and Salehi-Rad (2012).

Value

An object with S3 class "bmixt" is returned:

- **all_k**: A vector which includes the waiting times for all iterations. It is needed for monitoring the convergence of the BD-MCMC algorithm.
- **all_weights**: A vector which includes the waiting times for all iterations. It is needed for monitoring the convergence of the BD-MCMC algorithm.
- **pi_sample**: A vector which includes the MCMC samples after burn-in from parameter \(\pi\) of mixture distribution.
- **mu_sample**: A vector which includes the MCMC samples after burn-in from parameter \(\mu\) of mixture distribution.
- **sig_sample**: A vector which includes the MCMC samples after burn-in from parameter \(\sigma\) of mixture distribution.
- **data**: The original data.

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

References


See Also

`bmixnorm, bmixgamma, rmixt`

Examples

```r
## Not run:
# simulating data from mixture of normal with 3 components
n = 500
weight = c( 0.3, 0.5, 0.2 )
mean = c( 0 , 10 , 3 )
sd = c( 1 , 1 , 1 )

data = rmixnorm( n = n, weight = weight, mean = mean, sd = sd )

# plot for simulation data
hist( data, prob = TRUE, nclass = 30, col = "gray" )

x = seq( -20, 20, 0.05 )
densmixnorm = dmixnorm( x, weight, mean, sd )
lines( x, densmixnorm, lwd = 2 )

# running bdmcmc algorithm for the above simulation data set
bmixt.obj = bmixt( data, k = 3, iter = 1000 )

summary( bmixt.obj )

## End(Not run)
```

Galaxy data

This dataset considers 82 observations of the velocities (in 1000 km/second) of distant galaxies diverging from our own, from six well-separated conic sections of the Corona Borealis. The dataset has been analyzed under a variety of mixture models; See e.g. Stephens (2000).
mixgamma

Usage

data( galaxy )

Format

A data frame with 82 observations on the following variable.

speed  a numeric vector giving the speed of galaxies (in 1000 km/second).

References

Stephens, M. (2000) Bayesian analysis of mixture models with an unknown number of components-

Examples

data( galaxy )

hist( galaxy, prob = TRUE, xlim = c( 0, 40 ), ylim = c( 0, 0.3 ), nclass = 20,
     col = "gray", border = "white" )

lines( density( galaxy ), col = "black", lwd = 2 )

mixgamma  

*Mixture of Gamma distribution*

Description

Random generation and density function for the finite mixture of Gamma distribution.

Usage

rmixgamma( n = 10, weight = 1, alpha = 1, beta = 1 )

dmixgamma( x, weight = 1, alpha = 1, beta = 1 )

Arguments

n  The number of samples required.

x  The vector of quantiles.

weight  The vector of probability weights, with length equal to number of components (k). This is assumed to sum to 1; if not, it is normalized.

alpha  The vector of non-negative parameters of the Gamma distribution.

beta  The vector of non-negative parameters of the Gamma distribution.
Details

Sampling from finite mixture of Gamma distribution, with density:

\[
Pr(x|w, \alpha, \beta) = \sum_{i=1}^{k} w_i \Gamma(x|\alpha_i, \beta_i),
\]

where

\[
\Gamma(x|\alpha_i, \beta_i) = \frac{\beta_i^{\alpha_i} x^{\alpha_i-1} e^{-\beta_i x}}{\Gamma(\alpha_i)}.
\]

Value

Generated data as an vector with size \(n\).

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

References


See Also

rmixnorm, rmixt

Examples

```r
## Not run:
n = 10000
weight = c(0.6, 0.3, 0.1)
alpha = c(100, 200, 300)
beta = c(100/3, 200/4, 300/5)
data = rmixgamma(n = n, weight = weight, alpha = alpha, beta = beta)
hist(data, prob = TRUE, nclass = 30, col = "gray")

x = seq(-20, 20, 0.05)
densmixgamma = dmixnorm(x, weight, alpha, beta)
lines(x, densmixgamma, lwd = 2)
## End(Not run)
```
Description

Random generation and density function for the finite mixture of univariate Normal distribution.

Usage

```r
rmixnorm( n = 10L, weight = 1L, mean = 0L, sd = 1 )
dmixnorm( x, weight = 1, mean = 0, sd = 1 )
```

Arguments

- `n` The number of samples required.
- `x` The vector of quantiles.
- `weight` The vector of probability weights, with length equal to number of components (k). This is assumed to sum to 1; if not, it is normalized.
- `mean` The vector of means.
- `sd` The vector of standard deviations.

Details

Sampling from finite mixture of Normal distribution, with density:

\[ Pr(x|w, \mu, \sigma) = \sum_{i=1}^{k} w_i N(x|\mu_i, \sigma_i). \]

Value

Generated data as an vector with size `n`.

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

References


mixt

See Also

rmixt, rmixgamma

Examples

```r
## Not run:
n = 10000
weight = c( 0.3, 0.5, 0.2 )
mean = c( 0, 10, 3 )
sd = c( 1, 1, 1 )

data = rmixnorm( n = n, weight = weight, mean = mean, sd = sd )

hist( data, prob = TRUE, nclass = 30, col = "gray" )

x = seq( -20, 20, 0.05 )
densmixnorm = dmixnorm( x, weight, mean, sd )

lines( x, densmixnorm, lwd = 2 )
```

## End(Not run)

---

mixt  

*Mixture of t-distribution*

Description

Random generation and density function for the finite mixture of univariate t-distribution.

Usage

```r
rmixt( n = 10, weight = 1, df = 1, mean = 0, sd = 1 )

dmixt( x, weight = 1, df = 1, mean = 0, sd = 1 )
```

Arguments

- `n`: The number of samples required.
- `x`: The vector of quantiles.
- `weight`: The vector of probability weights, with length equal to number of components ($k$). This is assumed to sum to 1; if not, it is normalized.
- `df`: The vector of degrees of freedom ($> 0$, maybe non-integer). $df = Inf$ is allowed.
- `mean`: The vector of means.
- `sd`: The vector of standard deviations.
Details
Sampling from finite mixture of t-distribution, with density:

\[ Pr(x|w, \mu, \sigma) = \sum_{i=1}^{k} w_i N(x|\mu_i, \sigma_i). \]

Value
Generated data as an vector with size \( n \).

Author(s)
Reza Mohammadi <a.mohammadi@uva.nl>

References


See Also
rmixnorm, rmixgamma

Examples
```r
## not run:
n = 10000
weight = c( 0.3, 0.5, 0.2 )
df = c( 4, 4, 4 )
mean = c( 0, 10, 3 )
sd = c( 1, 1, 1 )
data = rmixt( n = n, weight = weight, df = df, mean = mean, sd = sd )
hist( data, prob = TRUE, nclass = 30, col = "gray" )
x = seq( -20, 20, 0.05 )
densmixt = dmixt( x, weight, df, mean, sd )
lines( x, densmixt, lwd = 2 )
## End(Not run)
```
### Description
Visualizes the results for function `bmixgamma`.

### Usage
```
## S3 method for class 'bmixgamma'
plot(x, ...)  
```

### Arguments
- **x**
  An object of S3 class "bmixgamma", from function `bmixgamma`.
- **...**
  System reserved (no specific usage).

### Author(s)
Reza Mohammadi `<a.mohammadi@uva.nl>`

### See Also
`bmixgamma`

### Examples
```
## Not run:
# simulating data from mixture of gamma with two components
n = 500  # number of observations
weight = c(0.6, 0.4)
alpha = c(12, 1)
beta = c(3, 2)
data <- rmixgamma(n = n, weight = weight, alpha = alpha, beta = beta)

# plot for simulation data
hist(data, prob = TRUE, nclass = 50, col = "gray")

x = seq(0, 10, 0.05)
truth = dmixgamma(x, weight, alpha, beta)
lines(x, truth, lwd = 2)

# Running bdmcmc algorithm for the above simulation data set
bmixgamma.obj <- bdmcmc(data)

plot(bmixgamma.obj)
## End(Not run)
```
plot.bmixnorm  Plot function for S3 class "bmixnorm"

Description
Visualizes the results for function `bmixnorm`.

Usage
```r
## S3 method for class 'bmixnorm'
plot(x, ...)  
```

Arguments
- `x`: An object of S3 class "bmixnorm", from function `bmixnorm`.
- `...`: System reserved (no specific usage).

Author(s)
Reza Mohammadi <a.mohammadi@uva.nl>

See Also
- `bmixnorm`

Examples
```r
## Not run:
# simulating data from mixture of Normal with 3 components
n   = 500
weight = c( 0.3, 0.5, 0.2 )
mean  = c( 0 , 10 , 3 )
sd    = c( 1 , 1 , 1 )
data = rmixnorm( n = n, weight = weight, mean = mean, sd = sd )

# plot for simulation data
hist( data, prob = TRUE, nclass = 30, col = "gray" )

x   = seq( -20, 20, 0.05 )
densmixnorm = dmixnorm( x, weight, mean, sd )
lines( x, densmixnorm, lwd = 2 )

# Running bdmcmc algorithm for the above simulation data set
bmixnorm.obj = bmixnorm( data, k = 3 )

plot( bmixnorm.obj )
## End(Not run)```
Description

Visualizes the results for function `bmixt`.

Usage

```r
## S3 method for class 'bmixt'
plot( x, ... )
```

Arguments

- `x`: An object of S3 class "bmixt", from function `bmixt`.
- `...`: System reserved (no specific usage).

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

See Also

`bmixt`

Examples

```r
## Not run:
# simulating data from mixture of Normal with 3 components
n = 500
weight = c( 0.3, 0.5, 0.2 )
mean = c( 0, 10, 3 )
sd = c( 1, 1, 1 )
data = rmixnorm( n = n, weight = weight, mean = mean, sd = sd )

# plot for simulation data
hist( data, prob = TRUE, nclass = 30, col = "gray" )

x = seq( -20, 20, 0.05 )
densmixnorm = dmixnorm( x, weight, mean, sd )
lines( x, densmixnorm, lwd = 2 )

# Running bdmcmc algorithm for the above simulation data set
bmixt.obj = bmixt( data, k = 3 )

plot( bmixt.obj )

## End(Not run)```
Description

Prints the information about the output of function `bmixgamma`.

Usage

```r
## S3 method for class 'bmixgamma'
print( x, ... )
```

Arguments

- `x`: An object of S3 class "bmixgamma", from function `bmixgamma`.
- `...`: System reserved (no specific usage).

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

See Also

- `bmixgamma`

Examples

```r
## Not run:
# simulating data from mixture of gamma with two components
n = 500 # number of observations
weight = c( 0.6, 0.4 )
alpha = c( 12, 1 )
beta = c( 3, 2 )
data <- rmixgamma( n = n, weight = weight, alpha = alpha, beta = beta )

# plot for simulation data
hist( data, prob = TRUE, nclass = 50, col = "gray" )

x = seq( 0, 10, 0.05 )
truth = dmixgamma( x, weight, alpha, beta )
lines( x, truth, lwd = 2 )

# Running bdmcmc algorithm for the above simulation data set
bmixgamma.obj <- bmixgamma( data, iter = 500 )

print( bmixgamma.obj )

## End(Not run)
```
print.bmixnorm

Print function for S3 class "bmixnorm"

Description
Prints the information about the output of function bmixnorm.

Usage

```r
## S3 method for class 'bmixnorm'
print( x, ... )
```

Arguments
- **x**: An object of S3 class "bmixnorm", from function bmixnorm.
- **...**: System reserved (no specific usage).

Author(s)
Reza Mohammadi <a.mohammadi@uva.nl>

See Also
- bmixnorm

Examples

```r
## Not run:
# simulating data from mixture of Normal with 3 components
n = 500
weight = c( 0.3, 0.5, 0.2 )
mean = c( 0, 10, 3 )
sd = c( 1, 1, 1 )
data = rmixnorm( n = n, weight = weight, mean = mean, sd = sd )

# plot for simulation data
hist( data, prob = TRUE, nclass = 30, col = "gray" )

x = seq( -20, 20, 0.05 )
densmixnorm = dmixnorm( x, weight, mean, sd )
lines( x, densmixnorm, lwd = 2 )

# running bdmcmc algorithm for the above simulation data set
bmixnorm.obj = bmixnorm( data, k = 3, iter = 1000 )
print( bmixnorm.obj )

## End(Not run)
```
print.bmixt

Print function for S3 class "bmixt"

Description

Prints the information about the output of function bmixt.

Usage

## S3 method for class 'bmixt'
print( x, ... )

Arguments

x  An object of S3 class "bmixt", from function bmixt.
... System reserved (no specific usage).

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

See Also

bmixt

Examples

## Not run:
# simulating data from mixture of Normal with 3 components
n = 500
weight = c( 0.3, 0.5, 0.2 )
mean = c( 0, 10, 3 )
sd = c( 1, 1, 1 )
data = rmixnorm( n = n, weight = weight, mean = mean, sd = sd )

# plot for simulation data
hist( data, prob = TRUE, nclass = 30, col = "gray" )

x   = seq( -20, 20, 0.05 )
densmixnorm = dmixnorm( x, weight, mean, sd )
lines( x, densmixnorm, lwd = 2 )

# running bdmcmc algorithm for the above simulation data set
bmixt.obj = bmixt( data, k = 3, iter = 1000 )

print( bmixt.obj )

## End(Not run)
rdirichlet  Random generation for the Dirichlet distribution

Description
Random generation from the Dirichlet distribution.

Usage
rdirichlet( n = 10L, alpha = c( 1L, 1 ) )

Arguments
n  The number of samples required.
alpha  The vector of shape parameters.

Details
The Dirichlet distribution is the multidimensional generalization of the beta distribution.

Value
A matrix with n rows, each containing a single Dirichlet random deviate.

Author(s)
Reza Mohammadi <a.mohammadi@uva.nl>

Examples
draws = rdirichlet( n = 500, alpha = c( 1, 1, 1 ) )
boxplot( draws )

summary.bmixgamma  Summary function for S3 class "bmixgamma"

Description
Provides a summary of the results for function bmixgamma.

Usage
## S3 method for class 'bmixgamma'
summary( object, ... )
Arguments

object

An object of S3 class "bmixgamma", from function bmixgamma.

... System reserved (no specific usage).

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

See Also

bmixgamma

Examples

## not run:
# simulating data from mixture of gamma with two components
n = 500 # number of observations
weight = c( 0.6, 0.4 )
alpha = c( 12 , 1 )
beta = c( 3 , 2 )

data <- rmixgamma( n = n, weight = weight, alpha = alpha, beta = beta )

# plot for simulation data
hist( data, prob = TRUE, nclass = 50, col = "gray" )

x = seq( 0, 10, 0.05 )
truth = dmixgamma( x, weight, alpha, beta )

lines( x, truth, lwd = 2 )

# Running bdmcmc algorithm for the above simulation data set
bmixgamma.obj <- bmixgamma( data, iter = 500 )

summary( bmixgamma.obj )

## End(not run)
**summary.bmixt**

**Arguments**

object

An object of S3 class "bmixnorm", from function `bmixnorm`.

... System reserved (no specific usage).

**Author(s)**

Reza Mohammadi <a.mohammadi@uva.nl>

**See Also**

`bmixnorm`

**Examples**

```r
## Not run:
# simulating data from mixture of normal with 3 components
n = 500
weight = c(0.3, 0.5, 0.2)
mean = c(0, 10, 3)
sd = c(1, 1, 1)
data = rmixnorm(n = n, weight = weight, mean = mean, sd = sd)

# plot for simulation data
hist(data, prob = TRUE, nclass = 30, col = "gray")

x = seq(-20, 20, 0.05)
densmixnorm = dmixnorm(x, weight, mean, sd)
lines(x, densmixnorm, lwd = 2)

# Running bdmmcmc algorithm for the above simulation data set
bmixnorm.obj = bmixnorm(data, k = 3, iter = 1000)
summary(bmixnorm.obj)

## End(Not run)
```

**Description**

Provides a summary of the results for function `bmixt`.

**Usage**

```r
## S3 method for class 'bmixt'
summary(object, ...)
```
Arguments

object An object of S3 class "bmixt", from function bmixt.

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

See Also

bmixt

Examples

```r
## not run:
# simulating data from mixture of normal with 3 components
n = 500
weight = c(0.3, 0.5, 0.2)
mean = c(0, 10, 3)
sd = c(1, 1, 1)
data = rmixnorm(n = n, weight = weight, mean = mean, sd = sd)

# plot for simulation data
hist(data, prob = TRUE, nclass = 30, col = "gray")

x = seq(-20, 20, 0.05)
densmixnorm = dmixnorm(x, weight, mean, sd)
lines(x, densmixnorm, lwd = 2)

# Running bdmcmc algorithm for the above simulation data set
bmixt.obj = bmixt(data, k = 3, iter = 1000)

summary(bmixt.obj)

## End(Not run)
```
Index

*Topic **Package, bmixture, Mixture distributions, Bayesian model selection**  
  bmixture-package, 2

*Topic **datasets**  
galaxy, 11

*Topic **distribution**  
rdirichlet, 23

  bmixnorm_fixed_k (bmixture-package), 2  
  bmixnorm_unknown_k  
    (bmixture-package), 2

  bmixgamma, 3, 5, 8, 11, 17, 20, 23, 24  
  bmixnorm, 5, 6, 11, 18, 21, 24, 25  
  bmixt, 5, 8, 9, 19, 22, 25, 26  
  bmixture-package, 2

  dmixgamma (mixgamma), 12  
  dmixnorm (mixnorm), 14  
  dmixt (mixt), 15

  galaxy, 11

  mixgamma, 12  
  mixnorm, 14  
  mixt, 15

  plot.bmixgamma, 17  
  plot.bmixnorm, 18  
  plot.bmixt, 19  
  print.bmixgamma, 20  
  print.bmixnorm, 21  
  print.bmixt, 22

  rdirichlet, 23  
  rmixgamma, 15, 16  
  rmixgamma (mixgamma), 12  
  rmixnorm, 8, 13, 16  
  rmixnorm (mixnorm), 14  
  rmixt, 11, 13, 15

rmixt (mixt), 15  
summary.bmixgamma, 23  
summary.bmixnorm, 24  
summary.bmixt, 25