Package ‘BayesMultiMode’

April 6, 2023

Type Package

Title Bayesian Mode Inference

Version 0.5.1

Description
A Bayesian approach for mode inference which works in two steps. First, a mixture distribution is fitted on the data using a sparse finite mixture (SFM) Markov chain Monte Carlo (MCMC) algorithm following Malsiner-Walli, Frühwirth-Schnatter and Grün (2016) <doi:10.1007/s11222-014-9500-2>). The number of mixture components does not have to be known; the size of the mixture is estimated endogenously through the SFM approach. Second, the modes of the estimated mixture at each MCMC draw are retrieved using algorithms specifically tailored for mode detection. These estimates are then used to construct posterior probabilities for the number of modes, their locations and uncertainties, providing a powerful tool for mode inference.

License GPL (>= 3)

Imports assertthat, bayesplot, dplyr, ggplot2, ggpubr, gtools, magrittr, MCMCglmm, mvtnorm, posterior, sn, stringr, tidyr, Rdpack, scales

Depends R (>= 3.5.0)

Suggests testthat (>= 3.0.0)

RdMacros Rdpack

Encoding UTF-8

LazyData true

URL https://github.com/paullabonne/BayesMultiMode

BugReports https://github.com/paullabonne/BayesMultiMode/issues

NeedsCompilation no

RoxygenNote 7.2.3

Config/testthat/edition 3

Author Nalan Baştürk [aut], Jamie Cross [aut], Peter de Knijff [aut],
Bayesian estimation of mixture distributions

Description

Gibbs samplers for sparse finite mixture Markov chain Monte Carlo (SFM MCMC) estimation.

Usage

bayes_estimation(
  data,
  K,
  dist,
  priors = list(),
  nb_iter = 2000,
  burnin = nb_iter/2,
  printing = TRUE
)
Arguments

data Vector of observations
K Maximum number of mixture components
dist String indicating the distribution of the mixture components Currently supports "normal", "skew_normal", "poisson" and "shifted_poisson"
priors List of priors; default is an empty list which implies the following priors:
a0 = 1,
A0 = 200,
b0 = median(y),
B0 = (max(y) - min(y))^2 (normal),
D_xi = 1,
D_psi =1, (skew normal: B0 = diag(D_xi,D_psi)),
c0 = 2.5,
l0 = 1.1 (poisson),
l0 = 5 (shifted poisson),
L0 = 1.1/median(y),
L0 = l0 - 1 (shifted poisson),
g0 = 0.5,
G0 = 100*g0/c0/B0 (normal),
G0 = g0/(0.5*var(y)) (skew normal)
b_n_iter Number of MCMC iterations; default is 2000
burnin Number of MCMC iterations used as burnin; default is nb_iter/2
printing Showing MCMC progression?

Value

A list of class ‘BayesMixture‘ containing

• data - Same as argument
• dist_type - Type of the distribution (continuous or discrete)
• pars_names - Names of the mixture components’ parameters
• mcmc - Matrix of MCMC draws where the rows corresponding to burnin have been discarded
• mcmc_all - Original matrix of MCMC draws

References


Frühwirth-Schnatter S, Malsiner-Walli G (2019). “From here to infinity: sparse finite versus Dirichlet process mixtures in model-based clustering.” Advances in Data Analysis and Classification, 13,
Examples

# Example with galaxy data
set.seed(123)
# retrieve galaxy data
y = galaxy
# estimation
bayesmix = bayes_estimation(data = y,
K = 5, #not many to run the example rapidly
dist = "normal",
nb_iter = 500, #not many to run the example rapidly
burnin = 100)
# plot estimated mixture
# plot(bayesmix, max_size = 200)

# Example with DNA data
set.seed(123)
# retrieve DNA data
y = d4z4
# estimation
bayesmix = bayes_estimation(data = y,
K = 5, #not many to run the example rapidly
dist = "shifted_poisson",
nb_iter = 500, #not many to run the example rapidly
burnin = 100)
# plot estimated mixture
# plot(bayesmix, max_size = 200)

bayes_mode

Bayesian mode inference

Description

Estimates modes for each mcmc draws which are then used to compute posterior probabilities for the number of modes and their locations. The fixed-point algorithm of Carreira-Perpinan (2000) is used for Gaussian mixtures while the Modal EM algorithm of Li et al. (2007) is used for other continuous mixtures.
bayes_mode

Usage

bayes_mode(
    BayesMix,
    rd = 1,
    tol_x = sd(BayesMix$data)/10,
    show_plot = FALSE,
    nb_iter = NULL
)

Arguments

BayesMix An object of class 'BayesMixture'
rd Rounding parameter
tol_x Tolerance parameter for distance in-between modes; default is sd(data)/10 where
data is an element of BayesMix. If two modes are closer than tol_x, only the
first estimated mode is kept. Not needed for mixtures of discrete distributions.
show_plot Show density with estimated mode as vertical bars ?
nb_iter Number of draws on which the mode-finding algorithm is run; default is NULL
which means the algorithm is run on all draws.

Value

A list of class ‘BayesMode’ containing

• data - from BayesMix argument
• dist - from BayesMix argument
• dist_type - from BayesMix argument
• pars_names - from BayesMix argument
• modes - Matrix with a row for each draw and columns showing modes
• p1 - Posterior probability of unimodality
• tb_nb_modes - Matrix showing posterior probabilities for the number of modes
• table_location - Matrix showing the posterior probabilities for location points being modes

References

Carreira-Perpinan MA (2000). “Mode-finding for mixtures of Gaussian distributions.” IEEE Trans-
actions on Pattern Analysis and Machine Intelligence, 22(11), 1318–1323. ISSN 1939-3539, doi:10.1109/
34.888716, Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

Examples

# Example with galaxy data
set.seed(123)

# retrieve galaxy data
y = galaxy

# estimation
bayesmix = bayes_estimation(data = y,
    K = 5, #not many to run the example rapidly
dist = "normal",
    nb_iter = 500, #not many to run the example rapidly
    burnin = 100)

# mode estimation
bayesmode = bayes_mode(bayesmix)

# plot
# plot(bayesmode, max_size = 200)

# summary
# summary(bayesmode)

# Example with DNA data
set.seed(123)

# retrieve galaxy data
y = d4z4

# estimation
bayesmix = bayes_estimation(data = y,
    K = 5, #not many to run the example rapidly
    dist = "shifted_poisson",
    nb_iter = 500, #not many to run the example rapidly
    burnin = 100)

# mode estimation
bayesmode = bayes_mode(bayesmix)

# plot
# plot(bayesmode, max_size = 200)

# summary
# summary(bayesmode)
**Description**

This is wrapper around the `mcmc_trace()` function from package bayesplot.

**Usage**

```r
bayes_trace(BayesMix, mcmc_vars = NULL, with_burnin = FALSE, ...)
```

**Arguments**

- **BayesMix**: An object of class BayesMixture
- **mcmc_vars**: Variables to plot; default is all the variable in the MCMC output
- **with_burnin**: Plot all draws?
- **...**: Additional arguments passed to function `mcmc_trace()` from the package bayesplot.

**Value**

A trace plot.

**Examples**

```r
# Example with galaxy data
set.seed(123)
# retrieve galaxy data
y = galaxy
# estimation
bayesmix = bayes_estimation(data = y, K = 5, # not many to run the example rapidly
dist = "normal",
nb_iter = 500, # not many to run the example rapidly
burnin = 100)
# trace plot
bayes_trace(bayesmix)
```

---

**ct47**

*X chromosomal macrosatellite repeats ct47*

**Description**

Repeat units that encode for a cancer testis antigen.

- **Locus (hg18): Xq24**
- **Unit (kb): 4.8**
- **Restriction enzyme: EcoRI**
- **Encoded product**: cancer testis antigen 47
Usage
cyclone
cT47

Format
A vector of counts with 410 elements.

References

cyclone

Tropical cyclones lifetime maximum intensity

Description
Dataset constructed using the International Best Track Archive for Climate Stewardship (IBTrACS). The distribution of tropical cyclones lifetime maximum intensity across the globe is known to be bimodal which has important implications for climate modelling.

Usage
cyclone

Format
A dataset with three columns showing the identification of the cyclone, its year of occurrence and its lifetime maximum intensity (LMI). LMI is calculated as the maximum wind speed for each cyclone with unit ks.

Source
https://www.ncei.noaa.gov/products/international-best-track-archive

References

**d4z4**

*Autosomal macrosatellite repeats d4z4*

**Description**

Macrosatellite repeats D4Z4 in the subtelomere of chromosome 4q.
Locus (hg18): 4q35.2
Unit (kb): 3.3
Restriction enzyme: EcoRI + HindIII/EcoRI + BlnI/XapI
Encoded product: DUX4

**Usage**

d4z4

**Format**

A vector of counts with 410 elements.

**References**


---

**discrete_MF**

*Mode-finding algorithm for mixture of discrete distributions*

**Description**

Mode-finding algorithm for mixture of discrete distributions

**Usage**

discrete_MF(
  mcmc,
  data,
  pars_names,
  dist = "NA",
  pmf_func = NULL,
  type = "all",
  show_plot = FALSE
)
Arguments

- **mcmc**: Vector of estimated mixture parameters
- **data**: Vector of observations used for estimating the mixture
- **pars_names**: Names of the mixture parameters; first element should correspond to the mixture proportions.
- **dist**: String indicating the distribution of the mixture components. Currently supports "poisson" and "shifted_poisson"; default is "NA"; only use this argument if you have used Poisson and shifted Poisson distributions identical to the one used in the package.
- **pmf_func**: Pmf of the mixture components associated with the mcmc draws (if mcmc estimation has not been carried out with BayesMultiMode); default is null
- **type**: Type of modes, either unique or all (the latter includes flat modes); default is "all"
- **show_plot**: If true show the data and estimated modes; default is false

Value

Vector of estimated modes

References


Examples

```r
# Example with the poisson distribution
lambda = c(0.1, 10)
p = c(0.5, 0.5)
params = c(eta = p, lambda = lambda)
pars_names = c("eta", "lambda")
dist = "poisson"
data = c(rpois(p[1]*1e3, lambda[1]), rpois(p[2]*1e3, lambda[2]))

modes = discrete_MF(params, data = data, pars_names = pars_names, dist = dist)

# Example with an arbitrary distribution
mu = c(20, 5)
size = c(20, 0.5)
p = c(0.5, 0.5)
params = c(eta = p, mu = mu, size = size)
pars_names = c("eta", "mu", "size")
data = c(rnbinom(p[1]*1e3, mu = mu[1], size = size[1]),
```

rnbinom(p[2]*1e3, mu = mu[2], size = size[2]))

pmf_func <- function(x, pars) {
  dbinom(x, mu = pars["mu"], size = pars["size"])
}

modes = discrete_MF(params, data = data, pars_names = pars_names, pmf_func = pmf_func)

---

fixed_point

Modal fixed-point algorithm

Description
Algorithm for estimating modes in mixture of Normal distributions.

Usage
fixed_point(mcmc, data, pars_names, tol_x = sd(data)/10, show_plot = F)

Arguments
mcmc Vector of estimated mixture parameters
data Vector of observations used for estimating the mixture
pars_names Names of the mixture parameters; first element should correspond to the mixture proportions; second to the mean; third to the standard deviation.
tol_x Tolerance parameter for distance in-between modes; default is sd(data)/10; if two modes are closer than tol_x, only the first estimated mode is kept.
show_plot If true show the data and estimated modes; default is false

Value
Vector of estimated modes

References

Examples
mu = c(0,5)
sigma = c(1,2)
p = c(0.5,0.5)
data = c(rnorm(p[1]*100, mu[1], sigma[1]), rnorm(p[2]*100, mu[2], sigma[2]))
params = c(eta = p, mu = mu, sigma = sigma)
pars_names = c("eta", "mu", "sigma")
modes = fixed_point(params, data, pars_names)

<table>
<thead>
<tr>
<th>galaxy</th>
<th>Galaxy series</th>
</tr>
</thead>
</table>

**Description**

Velocity at which 82 galaxies in the Corona Borealis region are moving away from our galaxy, scaled by 1000.

**Usage**

galaxy

**Format**

An object of class numeric of length 82.

**Source**

https://people.maths.bris.ac.uk/~mapjg/mixdata

**References**


---

**MEM**

*Modal EM algorithm (MEM)*

**Description**

Algorithm to find modes in mixture of continuous distributions.

**Usage**

MEM(
    mcmc,
    data,
    pars_names,
    dist = "NA",
    pdf_func = NULL,
    tol_x = sd(data)/10,
    show_plot = FALSE
)
MEM

Arguments

- **mcmc**: Vector of estimated mixture parameters
- **data**: Vector of observations used for estimating the mixture
- **pars_names**: Names of the mixture parameters; the first element of this vector should be the name of the mixture proportions. If you have used the skew normal of Azzalini, then the second element should correspond to the location, the third to the scale and the fourth to the shape.
- **dist**: String indicating the distribution of the mixture components; default is "NA". Currently supports "normal" and "skew_normal"; not needed if pdf_func is provided
- **pdf_func**: Ppdf of the mixture components associated with the mcmc draws (if mcmc estimation has not been carried out with BayesMultiMode); default is null
- **tol_x**: Tolerance parameter for distance in-between modes; default is sd(data)/10; if two modes are closer than tol_x, only the first estimated mode is kept.
- **show_plot**: If true show the data and estimated modes; default is false

Value

Vector of estimated modes

References


Examples

# Example with the skew normal =============================================
xi = c(0,6)
omega = c(1,2)
alpha = c(0,0)
p = c(0.8,0.2)
params = c(eta = p, xi = xi, omega = omega, alpha = alpha)
pars_names = c("eta", "xi", "omega", "alpha")
dist = "skew_normal"
data = c(sn::rsn(p[1]*100, xi[1], omega[1], alpha[1]), sn::rsn(p[2]*100, xi[2], omega[2], alpha[2]))
modes = MEM(params, data = data, pars_names = pars_names, dist = dist)

# Example with an arbitrary distribution =====================================
xi = c(0,6)
omega = c(1,2)
alpha = c(0,0)
nu = c(3,100)
p = c(0.8,0.2)
params = c(eta = p, mu = xi, sigma = omega, xi = alpha, nu = nu)
pars_names = c("eta", "mu", "sigma", "xi", "nu")

pdf_func <- function(x, pars) {
  sn::dst(x, pars["mu"], pars["sigma"], pars["xi"], pars["nu"])
}
data = c(sn::rst(p[1]*100, xi[1], omega[1], alpha[1], nu[1]),
        sn::rst(p[2]*100, xi[2], omega[2], alpha[2], nu[2]))

modes = MEM(params, pars_names = pars_names, data = data, pdf_func = pdf_func)

---
new_BayesMixture

Creating a S3 object of class ‘BayesMixture’

Description

Function for creating an object of class ‘bayes_estimation()’ which can subsequently be used as argument in ‘bayes_mode()’.

Usage

new_BayesMixture(
  mcmc,
  data,
  K,
  burnin,
  dist = "NA",
  pars_names,
  pdf_func = NULL,
  dist_type
)

Arguments

mcmc  A matrix of MCMC draws
data  A vector containing the data used for estimating the model and generating the MCMC draws
K     Number of mixture components
burnin Number of draws to discard as burnin
dist  Distribution family of the mixture components supported by the package (e.g. "normal", "student", "skew_normal" or "shifted_poisson")
Plot an estimated mixture for a given number of draws with an histogram of the data.

Usage

```r
## S3 method for class 'BayesMixture'
plot(x, max_size = 250, transparency = 0.1, ...)
```

Arguments

- **x**
  - An object of class BayesMixture.
- **max_size**
  - The number of MCMC draws to plot.
- **transparency**
  - Transparency of the density lines. Default is 0.1. Should be greater than 0 and below or equal to 1.
- **...**
  - Not used.
plot.BayesMode  

Plot Bayesian mode estimates.

Description

Plot Bayesian mode estimates.

Usage

```r
## S3 method for class 'BayesMode'
plot(x, graphs = c("p1", "number", "loc"), ...)
```

Arguments

- `x`: An object of class BayesMode.
- `graphs`: which plot to show ? Default is all three c("p1", "number", "loc").
- `...`: Not used.

summary.BayesMode

Summary of Bayesian mode estimates.

Description

Summary of Bayesian mode estimates.

Usage

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

Arguments

- `object`: An object of class BayesMode.
- `...`: Not used.
Index

* datasets
  ct47, 7
  cyclone, 8
  d4z4, 9
  galaxy, 12

bayes_estimation, 2
bayes_mode, 4
bayes_trace, 6

ct47, 7
cyclone, 8
d4z4, 9
discrete_MF, 9
fixed_point, 11
galaxy, 12

MEM, 12

new_BayesMixture, 14
plot.BayesMixture, 15
plot.BayesMode, 16
summary.BayesMode, 16