Package ‘dppmix’

May 8, 2026

Type Package

Title Determinantal Point Process Mixture Models

Version 0.1.2

Date 2026-02-25

Description Multivariate Gaussian mixture model with a determinant point process prior to promote the discovery of parsimonious components from observed data. See Xu, Mueller, Telesca (2016) <doi:10.1111/biom.12482>.

URL `https://bitbucket.org/djhshih/dppmix`

Imports stats, mvtnorm

Suggests devtools

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Author Yanxun Xu [aut],
Peter Mueller [aut],
Donatello Telesca [aut],
David J. H. Shih [aut, cre]

Maintainer David J. H. Shih <`djh.shih@gmail.com`>

Repository CRAN

Date/Publication 2026-02-25 09:20:02 UTC

Contents

`dgammapois` *Density function for Gamma-Poisson distribution.*

Description

Data follow the Poisson distribution parameterized by a mean parameter that follows a gamma distribution.

Usage

```
dgammapois(x, a, b = 1, log = FALSE)
```

Arguments

`x`	vector of x values
`a`	shape parameter for gamma distribution on mean parameter
`b`	rate parameter for gamma distribution on mean parameter
`log`	whether to return the density in log scale

Value

density values

`dppmix_mvnorm` *Fit a determinantal point process multivariate normal mixture model.*

Description

Discover clusters in multidimensional data using a multivariate normal mixture model with a determinantal point process prior.

Usage

```
dppmix_mvnorm(
  X,
  hparams = NULL,
  store = NULL,
  control = NULL,
  fixed = NULL,
  verbose = TRUE
)
```

Arguments

`X`	`N x J` data matrix of `N` observations and `J` features
`hparams`	a list of hyperparameter values: `delta`, `a0`, `b0`, `theta`, `sigma_prop_mu`
`store`	a vector of character strings specifying additional vars of interest; a value of `NA` indicates that samples of all parameters in the model will be stored
`control`	a list of control parameters: `niter`, `burnin`, `thin`
`fixed`	a list of fixed parameter values
`verbose`	whether to emit verbose message

Details

A determinantal point process (DPP) prior is a repulsive prior. Compare to mixture models using independent priors, a DPP mixutre model will often discover a parsimonious set of mixture components (clusters).

Model fitting is done by sampling parameters from the posterior distribution using a reversible jump Markov chain Monte Carlo sampling approach.

Given $X = [x_i]$, where each x_i is a D-dimensional real vector, we seek the posterior distribution the latent variable $z = [z_i]$, where each z_i is an integer representing cluster membership.

$$x_i \mid z_i \sim Normal(\mu_k, \Sigma_k)$$

$$z_i \sim Categorical(w)$$

$$w \sim Dirichlet([\delta ... \delta])$$

$$\mu_k \sim DPP(C)$$

where C is the covariance function that evaluates the distances among the data points:

$$C(x_1, x_2) = exp(-\sum_d \frac{(x_1 - x_2)^2}{\theta^2})$$

We also define $\Sigma_k = E_k \Lambda_k E_k^\top$, where E_k is an orthonormal matrix whose column represents eigenvectors. We further assume that $E_k = E$ is fixed across all cluster components so that E can be estimated as the eigenvectors of the covariance matrix of the data matrix X. Finally, we put a prior on the entries of the Λ_k diagonal matrix:

$$\lambda_{kd}^{-1} \sim Gamma(a_0, b_0)$$

Hence, the hyperameters of the model include: `delta`, `a0`, `b0`, `theta`, as well as sampling hyperparameter `sigma_pro_mu`, which controls the spread of the Gaussian proposal distribution for the random-walk Metropolis-Hastings update of the μ parameter.

The parameters (and their dimensions) in the model include: `K`, `z (N x 1)`, `w (K x 1)`, `lambda (K x J)`, `mu (K x J)`, `Sigma (J x J x K)`. If any parameter is fixed, then `K` must be fixed as well.

Value

a `dppmix_mcmc` object containing posterior samples of the parameters

References

Yanxun Xu, Peter Mueller, Donatello Telesca. Bayesian Inference for Latent Biologic Structure with Determinantal Point Processes. Biometrics. 2016;72(3):955-64.

Examples

```
set.seed(1)
ns <- c(3, 3)
means <- list(c(-6, -3), c(0, 4))
d <- rmvnorm_clusters(ns, means)

mcmc <- dppmix_mvnorm(d$X, verbose=FALSE)
res <- estimate(mcmc)
table(d$cl, res$z)
```

`estimate` *Estimate parameter.*

Description

Estimate parameter from fitted model.

Usage

```
estimate(object, pars, ...)
```

Arguments

`object`	fitted model
`pars`	names of parameters to estimate
`...`	other parameters to pass

`rbern` *Random generator for the Bernoulli distribution.*

Description

Random generator for the Bernoulli distribution.

Usage

```
rbern(n, prob)
```

Arguments

`n`	number of samples to generate
`prob`	event probability

Value

an `integer` vector of 0 (non-event) and 1 (event)

`rbvec` *Generate a random binary vector.*

Description

Generate a random binary vector.

Usage

```
rbvec(n, prob, e.min = 0)
```

Arguments

`n`	size of binary vector
`prob`	event probability (not accounting for minimum event constraint)
`e.min`	minimum number of events

Value

an `integer` vector of 0 and 1

`rdirichlet` *Random generator for the Dirichlet distribution.*

Description

Random generator for the Dirichlet distribution.

Usage

```
rdirichlet(n, alpha)
```

Arguments

`n`	number of vectors to generate
`alpha`	vector of parameters of the Dirichlet distribution

Value

a `matrix` in which each row vector is Dirichlet distributed

`rmvnorm_clusters` *Generate random multivarate clusters*

Description

Generate random multivarate clusters

Usage

```
rmvnorm_clusters(ns, means)
```

Arguments

`ns`	number of data points in each cluster
`means`	centers of each cluster

Value

list containing matrix `X` and labels `cl`

Examples

```
ns <- c(5, 8, 7)
means <- list(c(-6, 1), c(-1, -1), c(0, 4))
d <- rmvnorm_clusters(ns, means)
```

Index