Package 'DPP'

May 7, 2026

Type Package

Title Inference of Parameters of Normal Distributions from a Mixture of Normals

Version 0.1.2

Description This MCMC method takes a data numeric vector (Y) and assigns the elements of Y to a (potentially infinite) number of normal distributions. The individual normal distributions from a mixture of normals can be inferred.
Following the method described in Escobar (1994) <doi:10.2307/2291223> we use a Dirichlet Process Prior (DPP) to describe stochastically our prior assumptions about the dimensionality of the data.

License MIT + file LICENSE

Depends methods, Rcpp (>= 0.12.4), coda, stats

Suggests R.rsp

VignetteBuilder R.rsp

LinkingTo Rcpp

RcppModules DPPmcmc,Models

RoxygenNote 6.0.1

NeedsCompilation yes

Author Luis M. Avila [aut, cre],
Michael R. May [aut],
Jeff Ross-Ibarra [aut]

Maintainer Luis M. Avila <lmavila@gmail.com>

Repository CRAN

Date/Publication 2018-05-24 08:38:30 UTC

Contents

DPP-package *Inference of Parameters of Normal Distributions from a Mixture of Normals*

Description

This MCMC method takes a data numeric vector (Y) and assigns the elements of Y to a (potentially infinite) number of normal distributions. The individual normal distributions from a mixture of normals can be inferred. Following the method described in Escobar (1994) <doi:10.2307/2291223> we use a Dirichlet Process Prior (DPP) to describe stochastically our prior assumptions about the dimensionality of the data.

Details

DPP implements a Bayesian method to get a posterior probability for k normal distributions form a vector of n numeric values. We implemented an MCMC method as described in Escobar (1994). Using a Dirichlet process prior we describe stochastically our prior assumptions about the dimensionality of the data without specifying a fix number of clusters k, allowing us to infer the number of normal distributions or categories from a potentially infinite number of categories. DPP is implemented in C++ and made available to used within the R statistical environment using Rcpp (Eddelbuettel and Francois, 2011).

Author(s)

Luis M. Avila, Michael R. May, Jeff Ross-Ibarra

Maintainer: Luis M. Avila <lmavila@gmail.com>

References

Ferguson, Thomas S. A Bayesian analysis of some nonparametric problems. The annals of statistics (1973): 209-230.

Antoniak, Charles E. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. The Annals of Statistics (1974): 1152-1174.

Escobar, Michael D. Estimating Normal Means With a Dirichlet Process Prior. Journal of the American Statistical Association, 89(425), 1994.

Neal, Radford M. Markov chain sampling methods for Dirichlet process mixture models. Journal of Computational and Graphical Statistics 9.2 (2000): 249-265.

Eddelbuettel, Dirk and Romain Francois. Rcpp: Seamless R and C++ Integration. Journal Of Statistical Software, 40(8):1-18, 2011.

Examples

```
normal.model<-new(NormalModel,
                  mean_prior_mean=0.5,
                  mean_prior_sd=0.1,
                  sd_prior_shape=3,
                  sd_prior_rate=20,
                  estimate_concentration_parameter=TRUE,
                  concentration_parameter_alpha=10,
                  proposal_disturbance_sd=0.1)

 #simulating three normal distributions
 y <- c(rnorm(100,mean=0.2,sd=0.05), rnorm(100,0.7,0.05), rnorm(100,1.3,0.1))
 hist(y,breaks=30)

 #setwd("~/yourwd") #mcmc log files will be saved here
 my_dpp_analysis <- dppMCMC_C(data=y,
                              output = "output_prefix_",
                              model=normal.model,
                              num_auxiliary_tables=4,
                              expected_k=1.5,
                              power=1)
 #running the mcmc  , generations will be ignored because auto_stop=true
 ## Not run:
 my_dpp_analysis$run(generations=1000,auto_stop=TRUE,max_gen = 10000,min_ess = 500)

 #we get rid of the first 25% of the output (burn-in)
 hist(my_dpp_analysis$getNumCategoryTrace(0.25))
 my_dpp_analysis$getNumCategoryProbabilities(0.25)

## End(Not run)
```

---

dppMCMC_C *A Reference Class that provides DPP functionality*

Description

This class implements the main functionality of this package. The consturctor receives a numeric vector (Y) and priors. A model should be provided specifying the distributions to be used for inference (e.g. NormalModel for Normal distributions or GammaModel for Gamma distributions). Then an MCMC algorithm will be used to infer a number of distributions (k) that fit the data. The prior for the number of distributions is specified by the concentration_parameter_alpha and expected_k. Once the data and priors are specified the method run is used to start the inference.

Details

Class dppMCMC_C Class dppMCMC_C A Reference Class that provides DPP functionality

Fields

dpp_mcmc_object a DPPmcmc object

Methods

`getNumCategoryProbabilities(burnin_cutoff = 0.25)` returns the probabilities vector for inferred number of categories

`getNumCategoryTrace(burnin_cutoff = 0.25)` returns the trace vector for the inferred number of categories in the data

`initialize(data, output, model, num_auxiliary_tables = 4, expected_k = 2, power = 1, verbose = TRUE, sample` the class constructor, initializes DPPmcmc object with data and parameters

`run(generations, sample_freq = generations/1000, log_file, allocation_file, param_file, append = TRUE, r` starts the MCMC run

Examples

```
normal.model<-new(NormalModel,
                  mean_prior_mean=0.5,
                  mean_prior_sd=0.1,
                  sd_prior_shape=3,
                  sd_prior_rate=20,
                  estimate_concentration_parameter=TRUE,
                  concentration_parameter_alpha=10,
                   proposal_disturbance_sd=0.1)

#simulating three normal distributions
y <- c(rnorm(100,mean=0.2,sd=0.05), rnorm(100,0.7,0.05), rnorm(100,1.3,0.1))
hist(y,breaks=30)

#setwd("~/yourwd") #mcmc log files will be saved here
## Not run:
my_dpp_analysis <- dppMCMC_C(data=y,
                             output = "output_prefix_",
                             model=normal.model,
                             num_auxiliary_tables=4,
                             expected_k=1.5,
                             power=1)
#running the mcmc  , generations will be ignored because auto_stop=true
my_dpp_analysis$run(generations=1000,auto_stop=TRUE,max_gen = 10000,min_ess = 500)

#we get rid of the first 25% of the output (burn-in)
hist(my_dpp_analysis$getNumCategoryTrace(0.25))

my_dpp_analysis$getNumCategoryProbabilities(0.25)

## End(Not run)
```

`expectedNumberOfClusters`

Calculate the expected number of clusters from the number of individuals and a concentration parameter

---

Description

Calculate the expected number of clusters from the number of individuals and a concentration parameter

Usage

```
expectedNumberOfClusters(n, a)
```

Arguments

n	the number of individuals
a	the concentration parameter

Value

the expected number of clusters

Examples

```
expectedNumberOfClusters(100,0.2)
expectedNumberOfClusters(100,0.15)
```

GammaModel-class *Class* "GammaModel"

Description

Objects of the GammaModel class are initialized with prior parameters to be used by the MCMC algorithm in dppMCMC_C class. An object of the class GammaModel will be passed as an argument upon creation of the dppMCMC_C object that will run the MCMC code.

Methods

```
new(GammaModel,
                    shape_prior_mean=4,
                    shape_prior_sd=1,
                    rate_prior_mean=1.5,
                    rate_prior_sd=0.54,
                    estimate_concentration_parameter=TRUE,
                    concentration_parameter_alpha=10,
                    proposal_disturbance_sd=0.1)
      instantiates a GammaModel object

getParameters()
 returns a list with the parameters and arguments supplied upon object initialization
```

Examples

```
#creating an object of the class NormalModel
gamma.model<-new(GammaModel,
                    shape_prior_mean=4,
                    shape_prior_sd=1,
                    rate_prior_mean=1.5,
                    rate_prior_sd=0.54,
                    estimate_concentration_parameter=TRUE,
                    concentration_parameter_alpha=10,
                    proposal_disturbance_sd=0.1)
gamma.model$getParameters()
```

NormalModel-class *Class* "NormalModel"

Description

Objects of the NormalModel class are initialized with prior parameters to be used by the MCMC algorithm in dppMCMC_C class. An object of the class NormalMode will be passed as an argument upon creation of the dppMCMC_C object that will run the MCMC code.

Methods

```
new(NormalModel,
    mean_prior_mean=0.5,
    mean_prior_sd=0.1,
    sd_prior_shape=3,
    sd_prior_rate=20,
    estimate_concentration_parameter=TRUE,
    concentration_parameter_alpha=10,
    proposal_disturbance_sd=0.1)
       instantiates a NormalModel object

getParameters()
 returns a list with the parameters and arguments supplied upon object initialization
```

Examples

```
#creating an object of the class NormalModel
normal.model<-new(NormalModel,
                  mean_prior_mean=0.5,
                  mean_prior_sd=0.1,
                  sd_prior_shape=3,
                  sd_prior_rate=20,
                  estimate_concentration_parameter=TRUE,
                  concentration_parameter_alpha=10,
                  proposal_disturbance_sd=0.1)
normal.model$getParameters()
```

`simulateChineseRestaurant` *Simulate a discrete distribution as in the chinese restaurant problem*

Description

Simulate a discrete distribution as in the chinese restaurant problem

Usage

```
simulateChineseRestaurant(num_elements, chi)
```

Arguments

`num_elements`	the number of elements to be grouped
`chi`	the concentration parameter

Value

The sum of `x` and `y`

Examples

```
simulateChineseRestaurant(100, 0.2)
simulateChineseRestaurant(100, 0.8)
```

Index