Package 'dirichletprocess'

June 13, 2020

Type Package

Title Build Dirichlet Process Objects for Bayesian Modelling

Version 0.4.0

Maintainer Dean Markwick <dean.markwick@talk21.com>

Description Perform nonparametric Bayesian analysis using Dirichlet processes without the need to program the inference algorithms. Utilise included pre-built models or specify custom models and allow the 'dirichletprocess' package to handle the Markov chain Monte Carlo sampling. Our Dirichlet process objects can act as building blocks for a variety of statistical models including and not limited to: density estimation, clustering and prior distributions in hierarchical models. See Teh, Y. W. (2011) <https://www.stats.ox.ac.uk/~teh/research/npbayes/Teh2010a.pdf>, among many other sources.

Depends R (>= 2.10)

License GPL-3

Encoding UTF-8

LazyData true

Suggests testthat, knitr, rmarkdown, tidyr, dplyr

Imports gtools, ggplot2, mvtnorm

URL https://github.com/dm13450/dirichletprocess

BugReports https://github.com/dm13450/dirichletprocess/issues

RoxygenNote 6.1.1

NeedsCompilation no

Author Gordon J. Ross [aut],
Dean Markwick [aut, cre],
Kees Mulder [ctb] (<https://orcid.org/0000-0002-5387-3812>),
Giovanni Sighinolfi [ctb]

Repository CRAN

Date/Publication 2020-06-13 10:40:02 UTC

R topics documented:

`BetaMixture2Create` *Create a Beta mixture with zeros at the boundaries.*

Description

Create a Beta mixture with zeros at the boundaries.

Usage

```
BetaMixture2Create(priorParameters = 2, mhStepSize = c(1, 1),
  maxT = 1)
```

Arguments

`priorParameters`	The prior parameters for the base measure.
`mhStepSize`	The Metropolis Hastings step size. A numeric vector of length 2.
`maxT`	The upper bound of the Beta distribution. Defaults to 1 for the standard Beta distribution.

Value

A mixing distribution object.

`BetaMixtureCreate` *Create a Beta mixing distribution.*

Description

See `DirichletProcessBeta` for the default prior and hyper prior distributions.

Usage

```
BetaMixtureCreate(priorParameters = c(2, 8), mhStepSize = c(1, 1),
  maxT = 1, hyperPriorParameters = c(1, 0.125))
```

Arguments

priorParameters	The prior parameters for the base measure.
mhStepSize	The Metropolis Hastings step size. A numeric vector of length 2.
maxT	The upper bound of the Beta distribution. Defaults to 1 for the standard Beta distribution.
hyperPriorParameters	The parameters for the hyper prior.

Value

A mixing distribution object.

Burn *Add burn-in to a dirichletprocess object*

Description

Add burn-in to a dirichletprocess object

Usage

```
Burn(dpobj, niter)
```

Arguments

dpobj	A dirichletprocess object.
niter	Number of iterations to burn.

Value

A dirichletprocess object where all chain objects have the first `niter` iterations are removed.

Examples

```
dp <- Fit(DirichletProcessGaussian(rnorm(10)), 100)
DiagnosticPlots(dp)
burned_dp <- Burn(dp, 50)
DiagnosticPlots(burned_dp)
```

ChangeObservations *Change the observations of fitted Dirichlet Process.*

Description

Using a fitted Dirichlet process object include new data. The new data will be assigned to the best fitting cluster for each point.

Usage

```
ChangeObservations(dpobj, newData)
```

Arguments

dpobj	The Dirichlet process object.
newData	New data to be included

Value

Changed Dirichlet process object

Examples

```
y <- rnorm(10)
dp <- DirichletProcessGaussian(y)
dp <- ChangeObservations(dp, rnorm(10))
```

ClusterComponentUpdate *Update the component of the Dirichlet process*

Description

Update the cluster assignment for each data point.

Usage

```
ClusterComponentUpdate(dpObj)

## S3 method for class 'conjugate'
ClusterComponentUpdate(dpObj)

## S3 method for class 'hierarchical'
ClusterComponentUpdate(dpObj)
```

Arguments

dpObj	Dirichlet Process object

Value

Dirichlet process object with update components.

Examples

```
dp <- DirichletProcessGaussian(rnorm(10))
dp <- ClusterComponentUpdate(dp)
```

ClusterLabelPredict *Predict the cluster labels of some new data.*

Description

Given a fitted Dirichlet process object and some new data use this function to predict what clusters the new data belong to and associated cluster parameters.

Usage

```
ClusterLabelPredict(dpobj, newData)
```

Arguments

dpobj	Fitted Dirichlet Process
newData	New data to have cluster labels predicted.

Value

A list of the predicted cluster labels of some new unseen data.

Examples

```
y <- rnorm(10)
dp <- DirichletProcessGaussian(y)
dp <- Fit(dp, 5)
newY <- rnorm(10, 1)
pred <- ClusterLabelPredict(dp, newY)
```

ClusterParameterUpdate *Update the cluster parameters of the Dirichlet process.*

Description

Update the parameters of each individual cluster using all the data assigned to the particular cluster. A sample is taken from the posterior distribution using a direct sample if the mixing distribution is conjugate or the Metropolis Hastings algorithm for non-conjugate mixtures.

Usage

```
ClusterParameterUpdate(dpObj)
```

Arguments

dpObj	Dirichlet process object

Value

Dirichlet process object with update cluster parameters

Examples

```
dp <- DirichletProcessGaussian(rnorm(10))
dp <- ClusterParameterUpdate(dp)
```

DiagnosticPlots *Diagnostic plots for dirichletprocess objects*

Description

Plot several diagnostic plots for dirichletprocess objects. Because the dimension of the dirichletprocess mixture is constantly changing, it is not simple to create meaningful plots of the sampled parameters. Therefore, the plots focus on the likelihood, alpha, and the number of clusters.

Usage

```
DiagnosticPlots(dpobj, gg = FALSE)

AlphaTraceplot(dpobj, gg = TRUE)

AlphaPriorPosteriorPlot(dpobj, prior_color = "#2c7fb8",
  post_color = "#d95f02", gg = TRUE)
```

```
ClusterTraceplot(dpobj, gg = TRUE)

LikelihoodTraceplot(dpobj, gg = TRUE)
```

Arguments

dpobj	A dirichletprocess object that was fit.
gg	Logical; whether to create a ggplot or base R plot (if gg = FALSE). For DiagnosticPlots, this means that the plots will be given one-by-one, while base plots can be arranged in a grid.
prior_color	For AlphaPriorPosteriorPlot, the color of the prior function.
post_color	For AlphaPriorPosteriorPlot, the color of the posterior histogram.

Value

If gg = TRUE, a ggplot2 object. Otherwise, nothing is returned and a base plot is plotted.

Functions

- AlphaTraceplot: Trace plot of alpha.
- AlphaPriorPosteriorPlot: Plot of the prior and posterior of alpha.
- ClusterTraceplot: Trace plot of the number of clusters.
- LikelihoodTraceplot: Trace plot of the likelihood of the data for each iteration.

Examples

```
dp <- Fit(DirichletProcessGaussian(rnorm(10)), 100)
DiagnosticPlots(dp)
```

DirichletHMMCreate *Create a generic Dirichlet process hidden Markov Model*

Description

Create a hidden Markov model where the data is believed to be generated from the mixing object distribution.

Usage

```
DirichletHMMCreate(x, mdobj, alpha, beta)
```

Arguments

x	Data to be modelled
mdobj	Mixing disitribution object
alpha	Alpha parameter
beta	Beta parameter

dirichletprocess *A flexible package for fitting Bayesian non-parametric models.*

Description

Create, fit and take posterior samples from a Dirichlet process.

DirichletProcessBeta *Dirichlet process mixture of the Beta distribution.*

Description

Create a Dirichlet process object using the mean and scale parameterisation of the Beta distribution bounded on $(0, maxY)$.

Usage

```
DirichletProcessBeta(y, maxY, g0Priors = c(2, 8), alphaPrior = c(2, 4),
  mhStep = c(1, 1), hyperPriorParameters = c(1, 0.125),
  verbose = TRUE, mhDraws = 250)
```

Arguments

y	Data for which to be modelled.
maxY	End point of the data
g0Priors	Prior parameters of the base measure (α_0, β_0).
alphaPrior	Prior parameters for the concentration parameter. See also UpdateAlpha.
mhStep	Step size for Metropolis Hastings sampling algorithm.
hyperPriorParameters	Hyper-prior parameters for the prior distributions of the base measure parameters (a, b).
verbose	Logical, control the level of on screen output.
mhDraws	Number of Metropolis-Hastings samples to perform for each cluster update.

Details

$G_0(\mu, \nu|maxY, \alpha_0, \beta_0) = U(\mu|0, maxY)\text{Inv} - \text{Gamma}(\nu|\alpha_0, \beta_0)$.

The parameter β_0 also has a prior distribution $\beta_0 \sim \text{Gamma}(a, b)$ if the user selects `Fit(...,updatePrior=TRUE)`.

Value

Dirichlet process object

DirichletProcessBeta2 *Dirichlet process mixture of Beta distributions with a Uniform Pareto base measure.*

Description

Create a Dirichlet process object using the mean and scale parameterisation of the Beta distribution bounded on $(0, maxY)$. The Pareto distribution is used as a prior on the scale parameter to ensure that the likelihood is 0 at the boundaries.

Usage

```
DirichletProcessBeta2(y, maxY, g0Priors = 2, alphaPrior = c(2, 4),
  mhStep = c(1, 1), verbose = TRUE, mhDraws = 250)
```

Arguments

y	Data for which to be modelled.
maxY	End point of the data
g0Priors	Prior parameters of the base measure $(\gamma$.
alphaPrior	Prior parameters for the concentration parameter. See also UpdateAlpha.
mhStep	Step size for Metropolis Hastings sampling algorithm.
verbose	Logical, control the level of on screen output.
mhDraws	Number of Metropolis-Hastings samples to perform for each cluster update.

Details

$G_0(\mu, \nu|maxY, \alpha) = U(\mu|0, maxY)\text{Pareto}(\nu|x_m, \gamma)$.

Value

Dirichlet process object

DirichletProcessCreate *Create a Dirichlet Process object*

Description

Using a previously created Mixing Distribution Object (mdObject) create a Dirichlet process object.

Usage

```
DirichletProcessCreate(x, mdObject, alphaPriorParameters = c(1, 1),
  mhDraws = 250)
```

Arguments

x	Data
mdObject	Mixing Distribution Object
alphaPriorParameters	Prior parameters for the concentration parameter of the Dirichlet Process
mhDraws	Number of posterior samples to take in the nonconjugate case

---

DirichletProcessExponential

Create a Dirichlet Mixture of Exponentials

Description

This is the constructor function to produce a dirichletprocess object with a Exponential mixture kernel with unknown rate. The base measure is a Gamma distribution that is conjugate to the posterior distribution.

Usage

```
DirichletProcessExponential(y, g0Priors = c(0.01, 0.01),
  alphaPriors = c(2, 4))
```

Arguments

y	Data
g0Priors	Base Distribution Priors $\alpha_0, \beta_0)$
alphaPriors	Alpha prior parameters. See UpdateAlpha.

Details

$G_0(\theta|\alpha_0, \beta_0) = \text{Gamma}\left(\theta|\alpha_0, \beta_0\right)$

Value

Dirichlet process object

DirichletProcessGaussian

Create a Dirichlet Mixture of Gaussians

Description

This is the constructor function to produce a `dirichletprocess` object with a Gaussian mixture kernel with unknown mean and variance. The base measure is a Normal Inverse Gamma distribution that is conjugate to the posterior distribution.

Usage

```
DirichletProcessGaussian(y, g0Priors = c(0, 1, 1, 1),
  alphaPriors = c(2, 4))
```

Arguments

`y`	Data
`g0Priors`	Base Distribution Priors $\gamma = (\mu_0, k_0, \alpha_0, \beta_0)$
`alphaPriors`	Alpha prior parameters. See `UpdateAlpha`.

Details

$G_0(\theta|\gamma) = N\left(\mu|\mu_0, \frac{\sigma^2}{k_0}\right) \text{Inv} - \text{Gamma}\left(\sigma^2|\alpha_0, \beta_0\right)$

We recommend scaling your data to zero mean and unit variance for quicker convergence.

Value

Dirichlet process object

DirichletProcessHierarchicalBeta

Create a Hierarchical Dirichlet Mixture of Beta Distributions

Description

Create a Hierarchical Dirichlet Mixture of Beta Distributions

Usage

```
DirichletProcessHierarchicalBeta(dataList, maxY, priorParameters = c(2,
  8), hyperPriorParameters = c(1, 0.125), gammaPriors = c(2, 4),
  alphaPriors = c(2, 4), mhStepSize = c(0.1, 0.1), numSticks = 50,
  mhDraws = 250)
```

Arguments

dataList	List of data for each separate Dirichlet mixture object
maxY	Maximum value for the Beta distribution.
priorParameters	Prior Parameters for the top level base distribution.
hyperPriorParameters	Hyper prior parameters for the top level base distribution.
gammaPriors	Prior parameters for the top level concentration parameter.
alphaPriors	Prior parameters for the individual parameters.
mhStepSize	Metropolis Hastings jump size.
numSticks	Truncation level for the Stick Breaking formulation.
mhDraws	Number of Metropolis-Hastings samples to perform for each cluster update.

Value

dpobjlist A Hierarchical Dirichlet Process object that can be fitted, plotted etc.

DirichletProcessHierarchicalMvnormal2

Create a Hierarchical Dirichlet Mixture of semi-conjugate Multivariate Normal Distributions

---

Description

Create a Hierarchical Dirichlet Mixture of semi-conjugate Multivariate Normal Distributions

Usage

```
DirichletProcessHierarchicalMvnormal2(dataList, g0Priors,
  gammaPriors = c(2, 4), alphaPriors = c(2, 4), numSticks = 50,
  numInitialClusters = 1, mhDraws = 250)
```

Arguments

dataList	List of data for each separate Dirichlet mixture object
g0Priors	Prior Parameters for the top level base distribution.
gammaPriors	Prior parameters for the top level concentration parameter.
alphaPriors	Prior parameters for the individual parameters.
numSticks	Truncation level for the Stick Breaking formulation.
numInitialClusters	Number of clusters to initialise with.
mhDraws	Number of Metropolis-Hastings samples to perform for each cluster update.

Value

dpobjlist A Hierarchical Dirichlet Process object that can be fitted, plotted etc.

`DirichletProcessMvnormal` *Create a Dirichlet mixture of multivariate normal distributions.*

Description

$G_0(\boldsymbol{\mu}, \Lambda | \boldsymbol{\mu_0}, \kappa_0, \nu_0, T_0) = N(\boldsymbol{\mu} | \boldsymbol{\mu_0}, (\kappa_0 \Lambda)^{-1}) \mathrm{Wi}_{\nu_0}(\Lambda | T_0)$

Usage

```
DirichletProcessMvnormal(y, g0Priors, alphaPriors = c(2, 4),
  numInitialClusters = 1)
```

Arguments

`y`	Data
`g0Priors`	Prior parameters for the base distribution.
`alphaPriors`	Alpha prior parameters. See `UpdateAlpha`.
`numInitialClusters`	Number of clusters to initialise with.

`DirichletProcessMvnormal2` *Create a Dirichlet mixture of multivariate normal distributions with semi-conjugate prior.*

Description

Create a Dirichlet mixture of multivariate normal distributions with semi-conjugate prior.

Usage

```
DirichletProcessMvnormal2(y, g0Priors, alphaPriors = c(2, 4))
```

Arguments

`y`	Data
`g0Priors`	Prior parameters for the base distribution.
`alphaPriors`	Alpha prior parameters. See `UpdateAlpha`.

`DirichletProcessWeibull` *Create a Dirichlet Mixture of the Weibull distribution*

Description

The likelihood is parameterised as $\text{Weibull}(y|a,b) = \frac{a}{b}y^{a-1}\exp\left(-\frac{x^a}{b}\right)$. The base measure is a Uniform Inverse Gamma Distribution. $G_0(a,b|\phi,\alpha_0,\beta_0) = U(a|0,\phi)\text{Inv}-\text{Gamma}(b|\alpha_0,\beta_0)$ $\phi \sim \text{Pareto}(x_m,k)$ $\beta \sim \text{Gamma}(\alpha_0,\beta_0)$ This is a semi-conjugate distribution. The cluster parameter a is updated using the Metropolis Hastings algorithm an analytical posterior exists for b.

Usage

```
DirichletProcessWeibull(y, g0Priors, alphaPriors = c(2, 4),
  mhStepSize = c(1, 1), hyperPriorParameters = c(6, 2, 1, 0.5),
  verbose = FALSE, mhDraws = 250)
```

Arguments

`y`	Data.
`g0Priors`	Base Distribution Priors.
`alphaPriors`	Prior for the concentration parameter.
`mhStepSize`	Step size for the new parameter in the Metropolis Hastings algorithm.
`hyperPriorParameters`	Hyper prior parameters.
`verbose`	Set the level of screen output.
`mhDraws`	Number of Metropolis-Hastings samples to perform for each cluster update.

Value

Dirichlet process object

References

Kottas, A. (2006). Nonparametric Bayesian survival analysis using mixtures of Weibull distributions. Journal of Statistical Planning and Inference, 136(3), 578-596.

ExponentialMixtureCreate *Create a Exponential mixing distribution*

Description

See DirichletProcessExponential for details on the base measure.

Usage

```
ExponentialMixtureCreate(priorParameters = c(0.01, 0.01))
```

Arguments

priorParameters
: Prior parameters for the base measure.

Value

Mixing distribution object

Fit *Fit the Dirichlet process object*

Description

Using Neal's algorithm 4 or 8 depending on conjugacy the sampling procedure for a Dirichlet process is carried out. Lists of both cluster parameters, weights and the sampled concentration values are included in the fitted dpObj. When update_prior is set to TRUE the parameters of the base measure are also updated.

Usage

```
Fit(dpObj, its, updatePrior = FALSE, progressBar = TRUE)
```

Arguments

dpObj	Initialised Dirichlet Process object
its	Number of iterations to use
updatePrior	Logical flag, defaults to FAlSE. Set whether the parameters of the base measure are updated.
progressBar	Logical flag indicating whether to display a progress bar.

Value

A Dirichlet Process object with the fitted cluster parameters and labels.

References

Neal, R. M. (2000). Markov chain sampling methods for Dirichlet process mixture models. Journal of computational and graphical statistics, 9(2), 249-265.

`Fit.markov` *Fit a Hidden Markov Dirichlet Process Model*

Description

Fit a Hidden Markov Dirichlet Process Model

Usage

```
## S3 method for class 'markov'
Fit(dpObj, its, updatePrior = F, progressBar = F)
```

Arguments

`dpObj`	Initialised Dirichlet Process object
`its`	Number of iterations to use
`updatePrior`	Logical flag, defaults to `FAlSE`. Set whether the parameters of the base measure are updated.
`progressBar`	Logical flag indicating whether to display a progress bar.

Value

A Dirichlet Process object with the fitted cluster parameters and states.

`GaussianMixtureCreate` *Create a Normal mixing distribution*

Description

See `DirichletProcessGaussian` for details on the base measure.

Usage

```
GaussianMixtureCreate(priorParameters = c(0, 1, 1, 1))
```

Arguments

`priorParameters`	Prior parameters for the base measure.

Value

Mixing distribution object

`GlobalParameterUpdate` *Update the parameters of the hierarchical Dirichlet process object.*

Description

Update the parameters of the hierarchical Dirichlet process object.

Usage

```
GlobalParameterUpdate(dpobjlist)
```

Arguments

`dpobjlist`	List of Dirichlet Process objects.

`HierarchicalBetaCreate` *Create a Mixing Object for a hierarchical Beta Dirichlet process object.*

Description

Create a Mixing Object for a hierarchical Beta Dirichlet process object.

Usage

```
HierarchicalBetaCreate(n, priorParameters, hyperPriorParameters,
  alphaPrior, maxT, gammaPrior, mhStepSize, num_sticks)
```

Arguments

`n`	Number of data sets
`priorParameters`	The prior parameters for the top level base distribution.
`hyperPriorParameters`	Hyper prior parameters for the top level base distribution.
`alphaPrior`	Individual level concentration parameter priors.
`maxT`	Bounding value of the data.
`gammaPrior`	Concentration parameter for the top level priors.
`mhStepSize`	Metropolis Hastings step size for the posterior drawing.
`num_sticks`	Number of stick breaking values to use.

Value

A mixing distribution object.

`HierarchicalMvnormal2Create` *Create a Mixing Object for a hierarchical semi-conjugate Multivariate Normal Dirichlet process object.*

Description

Create a Mixing Object for a hierarchical semi-conjugate Multivariate Normal Dirichlet process object.

Usage

```
HierarchicalMvnormal2Create(n, priorParameters, alphaPrior, gammaPrior,
  num_sticks)
```

Arguments

`n`	Number of data sets
`priorParameters`	The prior parameters for the top level base distribution.
`alphaPrior`	Individual level concentration parameter priors.
`gammaPrior`	Concentration parameter for the top level priors.
`num_sticks`	Number of stick breaking values to use.

Value

A mixing distribution object.

`Initialise` *Initialise a Dirichlet process object*

Description

Initialise a Dirichlet process object by assigning all the data points to a single cluster with a posterior or prior draw for parameters.

Usage

```
Initialise(dpObj, posterior = TRUE, m = 3, verbose = TRUE,
  numInitialClusters = 1)
```

Arguments

dpObj	A Dirichlet process object.
posterior	TRUE/FALSE value for whether the cluster parameters should be from the posterior. If false then the values are from the prior.
m	Number of auxiliary variables to use for a non-conjugate mixing distribution. Defaults to m=3. See ClusterComponentUpdate for more details on m.
verbose	Logical flag indicating whether to output the acceptance ratio for non-conjugate mixtures.
numInitialClusters	Number of clusters to initialise with.

Value

A Dirichlet process object that has initial cluster allocations.

Likelihood.beta *Mixing Distribution Likelihood*

Description

Evaluate the Likelihood of some data x for some parameter θ.

Usage

```
## S3 method for class 'beta'
Likelihood(mdObj, x, theta)

## S3 method for class 'beta2'
Likelihood(mdObj, x, theta)

## S3 method for class 'exponential'
Likelihood(mdObj, x, theta)

Likelihood(mdObj, x, theta)

## S3 method for class 'mvnormal'
Likelihood(mdObj, x, theta)

## S3 method for class 'mvnormal2'
Likelihood(mdObj, x, theta)

## S3 method for class 'normal'
Likelihood(mdObj, x, theta)
```

Arguments

mdObj	Mixing Distribution
x	Data
theta	Parameters of distribution

Value

Likelihood of the data

LikelihoodDP *The likelihood of the Dirichlet process object*

Description

Calculate the likelihood of each data point with its parameter.

Usage

```
LikelihoodDP(dpobj)
```

Arguments

dpobj	The dirichletprocess object on which to calculate the likelihood.

LikelihoodFunction *The Likelihood function of a Dirichlet process object.*

Description

Collecting the fitted cluster parameters and number of datapoints associated with each parameter a likelihood can be calculated. Each cluster is weighted by the number of datapoints assigned.

Usage

```
LikelihoodFunction(dpobj, ind)
```

Arguments

dpobj	Dirichlet process object.
ind	The iteration number. Defaults to the last iteration.

Value

A function f(x) that represents the Likelihood of the dpobj.

Examples

```
y <- rnorm(10)
dp <- DirichletProcessGaussian(y)
dp <- Fit(dp, 5)
f <- LikelihoodFunction(dp)
plot(f(-2:2))
```

`MixingDistribution` *Create a mixing distribution object*

Description

The constructor function for a mixing distribution object. Use this function to prepare an object for use with the appropriate distribution functions.

Usage

```
MixingDistribution(distribution, priorParameters, conjugate,
  mhStepSize = NULL, hyperPriorParameters = NULL)
```

Arguments

`distribution`	The name of the distribution mixture
`priorParameters`	The prior parameters
`conjugate`	Whether the prior is conjugate to the Likelihood.
`mhStepSize`	The scale of the proposal parameter for the Metropolis Hastings algorithm. Not needed for conjugate mixtures.
`hyperPriorParameters`	Vector of hyperPriorParameters for the distribution.

`Mvnormal2Create` *Create a multivariate normal mixing distribution with semi conjugate prior*

Description

Create a multivariate normal mixing distribution with semi conjugate prior

Usage

```
Mvnormal2Create(priorParameters)
```

Arguments

`priorParameters`	The prior parameters for the Multivariate Normal.

`MvnormalCreate` *Create a multivariate normal mixing distribution*

Description

Create a multivariate normal mixing distribution

Usage

```
MvnormalCreate(priorParameters)
```

Arguments

`priorParameters`	The prior parameters for the Multivariate Normal.

`PenalisedLikelihood.beta` *Calculate the parameters that maximise the penalised likelihood.*

Description

Used to find suitable starting parameters for nonconjugate mixtures. For some mixing distributions this hasn't been implemented yet.

Usage

```
## S3 method for class 'beta'
PenalisedLikelihood(mdObj, x)

PenalisedLikelihood(mdObj, x)

## Default S3 method:
PenalisedLikelihood(mdObj, x)
```

Arguments

`mdObj`	Mixing distribution object
`x`	Data

`plot.dirichletprocess` *Plot the Dirichlet process object*

Description

For a univariate Dirichlet process plot the density of the data with the posterior distribution and credible intervals overlayed. For multivariate data the first two columns of the data are plotted with the data points coloured by their cluster labels. The additional arguments are not used for multivariate data.

Usage

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

plot_dirichletprocess_univariate(x, likelihood = FALSE, single = TRUE,
  data_fill = "black", data_method = "density", data_bw = NULL,
  ci_size = 0.05, xgrid_pts = 100, quant_pts = 100, xlim = NA)

plot_dirichletprocess_multivariate(x)
```

Arguments

`x`	Dirichlet Process Object to plot
`...`	Further arguments, currently ignored.
`likelihood`	Logical, indicating whether to plot the likelihood from the dpobj.
`single`	Logical, indicating whether to draw the posterior from the last iteration or use the full cluster sequence.
`data_fill`	Passed to ‘fill‘ in the data geom, for example a color. Defaults to "black".
`data_method`	A string containing either "density" (default), "hist"/"histogram", or "none". Data is plotted according to this method.
`data_bw`	Bandwith to be passed either as the binwidth of `geom_histogram`, or as the bw of `geom_density`.
`ci_size`	Numeric, the interval size to use. Defaults to .05.
`xgrid_pts`	Integer, the number of points on the x-axis to evaluate.
`quant_pts`	Integer, the number of posterior functions to use to obtain the posterior and its interval.
`xlim`	Default NA. If a vector of length two, the limits on the x-axis of the plot. If `NA` (default), the limits will be automatically chosen.

Value

A ggplot object.

Examples

```
dp <- DirichletProcessGaussian(c(rnorm(50, 2, .2), rnorm(60)))
dp <- Fit(dp, 100)
plot(dp)

plot(dp, likelihood = TRUE, data_method = "hist",
     data_fill = rgb(.5, .5, .8, .6), data_bw = .3)
```

`PosteriorClusters` *Generate the posterior clusters of a Dirichlet Process*

Description

Using the stick breaking representation the user can draw the posterior clusters and weights for a fitted Dirichlet Process. See also `PosteriorFunction`.

Usage

```
PosteriorClusters(dpobj, ind)
```

Arguments

`dpobj`	Fitted Dirichlet process
`ind`	Index for which the posterior will be drawn from. Defaults to the last iteration of the fit.

Value

A list with the weights and cluster parameters that form the posterior of the Dirichlet process.

Examples

```
y <- rnorm(10)
dp <- DirichletProcessGaussian(y)
dp <- Fit(dp, 5)
postClusters <- PosteriorClusters(dp)
```

`PosteriorDraw.exponential` *Draw from the posterior distribution*

Description

Draw from the posterior distribution

Usage

```
## S3 method for class 'exponential'
PosteriorDraw(mdObj, x, n = 1, ...)

PosteriorDraw(mdObj, x, n = 1, ...)

## S3 method for class 'mvnormal'
PosteriorDraw(mdObj, x, n = 1, ...)

## S3 method for class 'mvnormal2'
PosteriorDraw(mdObj, x, n = 1, ...)

## S3 method for class 'normal'
PosteriorDraw(mdObj, x, n = 1, ...)

## S3 method for class 'weibull'
PosteriorDraw(mdObj, x, n = 100, ...)
```

Arguments

`mdObj`	Mixing Distribution
`x`	Data
`n`	Number of draws
`...`	For a non-conjugate distribution the starting parameters. Defaults to a draw from the prior distribution.

Value

A sample from the posterior distribution

`PosteriorFrame` *Calculate the posterior mean and quantiles from a Dirichlet process object.*

Description

Calculate the posterior mean and quantiles from a Dirichlet process object.

Usage

```
PosteriorFrame(dpobj, xgrid, ndraws = 1000, ci_size = 0.1)
```

Arguments

`dpobj`	The Dirichlet process object to be drawn from.
`xgrid`	The x values the posterior is to be evaluated at.
`ndraws`	The number of posterior draws to take.
`ci_size`	The size of the credible interval draw in terms of percentage.

Value

A dataframe consisting of the posterior mean and credible intervals.

`PosteriorFunction` *Generate the posterior function of the Dirichlet function*

Description

Generate the posterior function of the Dirichlet function

Usage

```
PosteriorFunction(dpobj, ind)
```

Arguments

`dpobj`	Fitted Dirichlet Process object
`ind`	What iteration to draw the posterior function from. Defaults to the last iteration.

Value

A posterior function f(x).

Examples

```
y <- rnorm(10)
dp <- DirichletProcessGaussian(y)
dp <- Fit(dp, 5)
postFuncDraw <- PosteriorFunction(dp)
plot(-3:3, postFuncDraw(-3:3))
```

`PosteriorParameters` *Calculate the posterior parameters for a conjugate prior.*

Description

Calculate the posterior parameters for a conjugate prior.

Usage

```
PosteriorParameters(mdObj, x)

## S3 method for class 'mvnormal'
PosteriorParameters(mdObj, x)

## S3 method for class 'normal'
PosteriorParameters(mdObj, x)
```

Arguments

`mdObj`	Mixing distribution object
`x`	Data

Value

Parameters of the posterior distribution

`Predictive.exponential` *Calculate how well the prior predicts the data.*

Description

Calculate how well the prior predicts the data.

Usage

```
## S3 method for class 'exponential'
Predictive(mdObj, x)

Predictive(mdObj, x)

## S3 method for class 'mvnormal'
Predictive(mdObj, x)

## S3 method for class 'normal'
Predictive(mdObj, x)
```

Arguments

mdObj	The distribution
x	The data

Value

The probability of the data being from the prior.

---

print.dirichletprocess

Print the Dirichlet process object

Description

Print a Dirichlet process object. This will print some basic information about the `dirichletprocess` object.

Usage

```
## S3 method for class 'dirichletprocess'
print(x, param_summary = FALSE, digits = 2,
  ...)
```

Arguments

x	Dirichlet Process Object to print.
param_summary	If `TRUE`, print the overall averages of each parameter of the model. Note that this averages over all clusters and over all iterations, so it will only give a loose sense of the resulting DPM model.
digits	Integer; Number of digits to display.
...	Further arguments passed to or from other methods.

Examples

```
dp <- Fit(DirichletProcessGaussian(rnorm(10)), 100)
dp
```

`PriorDensity.beta` *Calculate the prior density of a mixing distribution*

Description

Calculate the prior density of a mixing distribution

Usage

```
## S3 method for class 'beta'
PriorDensity(mdObj, theta)

## S3 method for class 'beta2'
PriorDensity(mdObj, theta)

PriorDensity(mdObj, theta)

## S3 method for class 'weibull'
PriorDensity(mdObj, theta)
```

Arguments

`mdObj`	Mixing distribution
`theta`	Prior parameters

`PriorDraw.beta` *Draw from the prior distribution*

Description

Draw from the prior distribution

Usage

```
## S3 method for class 'beta'
PriorDraw(mdObj, n = 1)

## S3 method for class 'beta2'
PriorDraw(mdObj, n = 1)

## S3 method for class 'exponential'
PriorDraw(mdObj, n)

PriorDraw(mdObj, n)

## S3 method for class 'mvnormal'
PriorDraw(mdObj, n = 1)

## S3 method for class 'mvnormal2'
PriorDraw(mdObj, n = 1)

## S3 method for class 'normal'
PriorDraw(mdObj, n = 1)

## S3 method for class 'weibull'
PriorDraw(mdObj, n = 1)
```

Arguments

mdObj	Mixing Distribution
n	Number of draws.

Value

A sample from the prior distribution

PriorParametersUpdate.beta

Update the prior parameters of a mixing distribution

Description

Update the prior parameters of a mixing distribution

Usage

```
## S3 method for class 'beta'
PriorParametersUpdate(mdObj, clusterParameters, n = 1)
```

```
PriorParametersUpdate(mdObj, clusterParameters, n = 1)

## S3 method for class 'weibull'
PriorParametersUpdate(mdObj, clusterParameters, n = 1)
```

Arguments

mdObj	Mixing Distribution Object
clusterParameters	Current cluster parameters
n	Number of samples

Value

mdobj New Mixing Distribution object with updated cluster parameters

---

rats *Tumour incidences in rats*

Description

Rat tumour data from Tarone (1982). Data from Table 5.1 of Bayesian Data Analysis

Usage

```
rats
```

Format

y number of rats with a tumour

N total number of rats in the experiment

Source

http://www.stat.columbia.edu/~gelman/book/data/rats.asc

`StickBreaking` *The Stick Breaking representation of the Dirichlet process.*

Description

A Dirichlet process can be represented using a stick breaking construction

$$G = \sum_{i=1}^{n} pi_i \delta_{\theta_i}$$

, where $\pi_k = \beta_k \prod_{k=1}^{n-1}(1 - \beta_k)$ are the stick breaking weights. The atoms δ_{θ_i} are drawn from G_0 the base measure of the Dirichlet Process. The $\beta_k \sim \mathrm{Beta}(1, \alpha)$. In theory n should be infinite, but we chose some value of N to truncate the series. For more details see reference.

Usage

```
StickBreaking(alpha, N)

piDirichlet(betas)
```

Arguments

`alpha`	Concentration parameter of the Dirichlet Process.
`N`	Truncation value.
`betas`	Draws from the Beta distribution.

Value

Vector of stick breaking probabilities.

Functions

- `piDirichlet`: Function for calculating stick lengths.

References

Ishwaran, H., & James, L. F. (2001). Gibbs sampling methods for stick-breaking priors. Journal of the American Statistical Association, 96(453), 161-173.

`true_cluster_labels` *Identifies the correct clusters labels, in any dimension, when cluster parameters and global parameters are matched.*

Description

Identifies the correct clusters labels, in any dimension, when cluster parameters and global parameters are matched.

Usage

```
true_cluster_labels(array, dpObj)
```

Arguments

`array`	The array containing matching indexes.
`dpObj`	A hierarchical dirichletprocess object.

Value

The array containing the correct matching indexes

`UpdateAlpha` *Update the Dirichlet process concentration parameter.*

Description

Using the procedure outlined in West (1992) we sample the concentration parameter of the Dirichlet process. See reference for further details.

Usage

```
UpdateAlpha(dpobj)

## Default S3 method:
UpdateAlpha(dpobj)

## S3 method for class 'hierarchical'
UpdateAlpha(dpobj)
```

Arguments

`dpobj`	Dirichlet process object.

Value

A Dirichlet process object with updated concentration parameter.

References

West, M. (1992). Hyperparameter estimation in Dirichlet process mixture models. ISDS Discussion Paper# 92-A03: Duke University.

`UpdateAlphaBeta` *Update the α and β parameter of a hidden Markov Dirichlet process model.*

Description

Update the α and β parameter of a hidden Markov Dirichlet process model.

Usage

```
UpdateAlphaBeta(dp)
```

Arguments

`dp`	Dirichlet process object

`WeibullMixtureCreate` *Create a Weibull mixing distribution.*

Description

See `DirichletProcessWeibull` for the default prior and hyper prior distributions.

Usage

```
WeibullMixtureCreate(priorParameters, mhStepSize,
  hyperPriorParameters = c(6, 2, 1, 0.5))
```

Arguments

`priorParameters`	Prior parameters for the Weibull parameters
`mhStepSize`	Metropolis Hastings Step Size
`hyperPriorParameters`	Parameters for the hyper-priors

Value

A mixing distribution object.

`weighted_function_generator` *Generate a weighted function.*

Description

Generate a weighted function.

Usage

```
weighted_function_generator(func, weights, params)
```

Arguments

`func`	Function that is used of the form func(x, params).
`weights`	Weighting of each cluster.
`params`	Cluster parameter list

Value

weighted function

Index