Package ‘BayesVarSel’

February 19, 2015

Type Package
Title Bayes Factors, Model Choice And Variable Selection In Linear Models
Version 1.6.1
Date 2015-01-01
Author Gonzalo Garcia-Donato and Anabel Forte
Maintainer Anabel Forte <anabel.forte@uv.es>

Description
Conceived to calculate Bayes factors in linear models and then to provide a formal Bayesian answer to testing and variable selection problems. From a theoretical side, the emphasis in the package is placed on the prior distributions and BayesVarSel allows using a wide range of them: Jeffreys-Zellner-Siow (Jeffreys, 1961; Zellner and Siow, 1980,1984) Zellner (1986); Fernandez et al. (2001), Liang et al. (2008) and Bayarri et al. (2012). The interaction with the package is through a friendly interface that syntactically mimics the well-known lm command of R. The resulting objects can be easily explored providing the user very valuable information (like marginal, joint and conditional inclusion probabilities of potential variables; the highest posterior probability model, HPM; the median probability model, MPM) about the structure of the true -data generating- model. Additionally, BayesVarSel incorporates abilities to handle problems with a large number of potential explanatory variables through parallel and heuristic versions (Garcia-Donato and Martinez-Beneito 2013) of the main commands.

Depends R (>= 3.0), parallel, MASS
License GPL-2
NeedsCompilation yes
Repository CRAN
Date/Publication 2015-01-26 20:32:02

R topics documented:

BayesVarSel-package ................................................. 2
BayesFactor ......................................................... 3
Bvs ............................................................... 6
GibbsBvs ............................................................ 9
Hypothesis testing, model selection and model averaging are important statistical problems that have in common the explicit consideration of the uncertainty about which is the true model. The formal Bayesian tool to solve such problems is the Bayes factor (Kass and Raftery, 1995) that reports the evidence in the data favoring each of the entertained hypotheses/models and can be easily translated to posterior probabilities.

This package has been specifically conceived to calculate Bayes factors in linear models and then to provide a formal Bayesian answer to testing and variable selection problems. From a theoretical side, the emphasis in the package is placed on the prior distributions (a very delicate issue in this context) and BayesVarSel allows using a wide range of them: Jeffreys-Zellner-Siow (Jeffreys, 1961; Zellner and Siow, 1980,1984) Zellner (1986); Fernandez et al. (2001), Liang et al. (2008) and Bayarri et al. (2012).

The interaction with the package is through a friendly interface that syntactically mimics the well-known lm command of R. The resulting objects can be easily explored providing the user very valuable information (like marginal, joint and conditional inclusion probabilities of potential variables; the highest posterior probability model, HPM; the median probability model, MPM) about the structure of the true -data generating- model. Additionally, BayesVarSel incorporates abilities to handle problems with a large number of potential explanatory variables through parallel and heuristic versions (Garcia-Donato and Martinez-Beneito 2013) of the main commands.

Author(s)

Gonzalo Garcia-Donato and Anabel Forte
Maintainer: Anabel Forte <anabel.forte@uv.es>
References


See Also

BayesFactor, Bvs, PBvs, GibbsBvs

Examples

demo(BayesVarSel.Hald)

BayesFactor and posterior probabilities for linear regression models

Description

Computes the Bayes factors and posterior probabilities of a list of linear regression models proposed to explain a common response variable over the same dataset

Usage

BayesFactor(models, data, prior.betas = "Robust", prior.models = "Constant", priorprobs=NULL)
Arguments

models  A named list with the entertained models defined with their corresponding formulas. One model must be nested in all the others.
data  data frame containing the data.
prior.betas  Prior distribution for regression parameters within each model. Possible choices include "Robust", "Liangetal", "gZellner", "ZellnerSiow" and "FLS" (see details).
prior.models  Prior probabilities of the models. Possible choices are "Constant" and "User" (see details).
priorprobs  A named list (same length and names as in argument models) with the prior probabilities of the models.)

Details

The Bayes factors, Bi, are expressed in relation with the simplest model (the one nested in all the others). Then, the posterior probabilities of the entertained models are obtained as
\[
Pr(M_i | \text{data})=Pr(M_i)*Bi/C,
\]
where Pr(Mi) is the prior probability of model Mi and C is the normalizing constant.

The Bayes factor B_i depends on the prior assigned for the regression parameters in Mi. BayesFactor implements a number of popular choices plus the "Robust" prior recently proposed by Bayarri et al (2012). The "Robust" prior is the default choice for both theoretical (see the reference for details) and computational reasons since it produces Bayes factors with closed-form expressions. The "gZellner" prior implemented corresponds to the prior in Zellner (1986) with g=n while the "Liangetal" prior is the hyper-g/n with a=3 (see the original paper Liang et al 2008, for details). "ZellnerSiow" is the multivariate Cauchy prior proposed by Zellner and Siow (1980, 1984), further studied by Bayarri and Garcia-Donato (2007). Finally, "FLS" is the prior recommended by Fernandez, Ley and Steel (2001) which is the prior in Zellner (1986) with g=\max(n, p*p) p being the difference between the dimension of the most complex model and the simplest one.

With respect to the prior over the model space Pr(Mi) three possibilities are implemented: "Constant", under which every model has the same prior probability and "User". With this last option, the prior probabilities are defined through the named list priorprobs. These probabilities can be given unnormalized.

Limitations: the error "A Bayes factor is infinite.". Bayes factors can be extremely big numbers if i) the sample size is even moderately large or if ii) a model is much better (in terms of fit) than the model taken as the null model. We are currently working on more robust implementations of the functions to handle these problems. In the meanwhile you could try using the g-Zellner prior (which is the most simple one and results, in these cases, should not vary much with the prior) and/or using more accurate definitions of the simplest model.

Value

BayesFactor returns an object of type BayesFactor which is a list with the following elements:

BFio  A vector with the Bayes factor of each model to the simplest model.
PostProbi  A vector with the posterior probabilities of each model.
BayesFactor

models A list with the entertained models.
nullmodel The position of the simplest model.

Author(s)

Gonzalo Garcia-Donato and Anabel Forte
Maintainer: <anabel.forte@uv.es>

References


See Also

bvs for variable selection within linear regression models

Examples

```r
## Not run:
#Analysis of Crime Data
#load data
data(UScrime)
#Model selection among the following models: (note model1 is nested in all the others)
model1<- as.formula("y~1+Prob")
model2<- as.formula("y~1+Prob+Time")
model3<- as.formula("y~1+Prob+Poli+Pol2")
model4<- as.formula("y~1+Prob+So")
model5<- as.formula("y~.")

#Equal prior probabilities for models:
crime.BF<- BayesFactor(models=list(basemodel=model1,
```
# Another configuration of prior probabilities of models:
crime.BF2 <- BayesFactor(models=list(basemodel=model1, ProbTimemodel=model2, ProbPolmodel=model3, ProbSomodel=model4, fullmodel=model5), data=UScrime, prior.models = "User", priorprobs=list(basemodel=1/8, ProbTimemodel=1/2, ProbPolmodel=1/8, ProbSomodel=1/8, fullmodel=1/8))
# same as:
# crime.BF2 <- BayesFactor(models=list(basemodel=model1, ProbTimemodel=model2, ProbPolmodel=model3, ProbSomodel=model4, fullmodel=model5), data=UScrime, 
# prior.models = "User", priorprobs=list(basemodel=1, ProbTimemodel=1, 
# ProbPolmodel=4, #ProbSomodel=1, fullmodel=1))

## End(Not run)

---

**Bvs**

Bayesian Variable Selection for linear regression models

**Description**

Exact computation of summaries of the posterior distribution using sequential computation.

**Usage**

```r
Bvs(formula, fixed.cov=c("Intercept"), data, prior.betas = "Robust", 
    prior.models = "Constant", n.keep = 10, time.test = TRUE, priorprobs=NULL)
```

**Arguments**

- **formula**: Formula defining the most complex regression model in the analysis. See details.
- **fixed.cov**: A character vector with the names of the covariates that will be considered as fixed (no variable selection over these). This argument provides an implicit definition of the simplest model considered. Default is "Intercept". Use "Null" if selection should be performed over all the variables defined by formula.
- **data**: data frame containing the data.
- **prior.betas**: Prior distribution for regression parameters within each model. Possible choices include "Robust", "Liangetal", "gZellner", "ZellnerSiow" and "FLS" (see details).
- **prior.models**: Prior distribution over the model space. Possible choices are "Constant", "ScottBerger" and "User" (see details).
- **n.keep**: How many of the most probable models are to be kept? By default is set to 10, which is automatically adjusted if 10 is greater than the total number of models.
- **time.test**: If TRUE and the number of variables is moderately large (>=18) a preliminary test to estimate computational time is performed.
priorprobs  A p+1 (p is the number of non-fixed covariates) dimensional vector defining the prior probabilities \( \Pr(M_i) \) (should be used in the case where prior.models="User"; see details.)

Details

The model space is the set of all models, \( M_i \), that contain the intercept and are nested in that specified by \texttt{formula}. The simplest of such models, \( M_0 \), contains only the intercept. Then \texttt{bvs} provides exact summaries of the posterior distribution over this model space, that is, summaries of the discrete distribution which assigns to each model \( M_i \) its probability given the data:

\[
\Pr(M_i | \text{data}) = \Pr(M_i) \ast B_i / C,
\]

where \( B_i \) is the Bayes factor of \( M_i \) to \( M_0 \), \( \Pr(M_i) \) is the prior probability of \( M_i \) and \( C \) is the normalizing constant.

The Bayes factor \( B_i \) depends on the prior assigned for the regression parameters in \( M_i \) and \texttt{bvs} implements a number of popular choices plus the "Robust" prior recently proposed by Bayarri et al (2012). The "Robust" prior is the default choice for both theoretical (see the reference for details) and computational reasons since it produces Bayes factors with closed-form expressions. The "gZellner" prior implemented corresponds to the prior in Zellner (1986) with \( g=n \) while the "Liangetal" prior is the hyper-\( g/h \) with \( a=3 \) (see the original paper Liang et al 2008, for details).

"ZellnerSiow" is the multivariate Cauchy prior proposed by Zellner and Siow (1980, 1984), further studied by Bayarri and Garcia-Donato (2007). Finally, "FLS" is the prior recommended by Fernandez, Ley and Steel (2008) which is the prior in Zellner (1986) with \( g=\max(n, p^p) \) \( p \) being the number of covariates to choose from (the most complex model has \( p+\)number of fixed covariates).

With respect to the prior over the model space \( \Pr(M_i) \) three possibilities are implemented: "Constant", under which every model has the same prior probability, "ScottBerger" under which \( \Pr(M_i) \) is inversely proportional to the number of models of that dimension, and "User" (see below). The "ScottBerger" prior was studied by Scott and Berger (2010) and controls for multiplicity.

When the parameter \texttt{prior.models="User"}, the prior probabilities are defined through the p+1 dimensional parameter vector \texttt{priorprobs}. Let \( k \) be the number of explanatory variables in the simplest model (the one defined by \texttt{fixed.cov}) then except for the normalizing constant, the first component of \texttt{priorprobs} must contain the probability of each model with \( k \) covariates (there is only one); the second component of \texttt{priorprobs} should contain the probability of each model with \( k+1 \) covariates and so on. Finally, the p+1 component in \texttt{priorprobs} defined the probability of the most complex model (that defined by \texttt{formula}). That is

\[
\texttt{priorprobs[j]} = C \ast \Pr(M_i \text{ such that } M_i \text{ has } j-l+k \text{ explanatory variables})
\]

where \( C \) is the normalizing constant, i.e \( C=1/\text{sum(priorprobs*choose(p,0:p))} \).

Note that \texttt{prior.models="Constant"} is equivalent to the combination \texttt{prior.models="User"} and \texttt{priorprobs=rep(1,(p+1))} but the internal functions are not the same and you can obtain small variations in results due to these differences in the implementation.

Similarly, \texttt{prior.models = "ScottBerger"} is equivalent to the combination \texttt{prior.models="User"} and \texttt{priorprobs = 1/choose(p,0:p)}.

Limitations: the error "A Bayes factor is infinite.". Bayes factors can be extremely big numbers if i) the sample size is even moderately large or if ii) a model is much better (in terms of fit) than the model taken as the null model. We are currently working on more robust implementations of the functions to handle these problems. In the meanwhile you could try using the g-Zellner prior (which
is the most simple one and results, in these cases, should not vary much with the prior) and/or using more accurate definitions of the simplest model (via the fixed.cov argument).

**Value**

`Bvs` returns an object of class `Bvs` with the following elements:

- **time**: The internal time consumed in solving the problem.
- **lmfull**: The `lm` class object that results when the model defined by `formula` is fitted by `lm`.
- **lmnull**: The `lm` class object that results when the model defined by `fixed.cov` is fitted by `lm`.
- **variables**: The name of all the potential explanatory variables (the set of variables to select from).
- **n**: Number of observations.
- **p**: Number of explanatory variables to select from.
- **k**: Number of fixed variables.
- **HPMbin**: The binary expression of the Highest Posterior Probability model.
- **modelsprob**: A `data.frame` which summaries the `n.keep` most probable, a posteriori models, and their associated probability.
- **inclprob**: A `data.frame` with the inclusion probabilities of all the variables.
- **jointinclprob**: A `data.frame` with the joint inclusion probabilities of all the variables.
- **postprobdim**: Posterior probabilities of the dimension of the true model.
- **call**: The call to the function.
- **method**: `full`.

**Author(s)**

Gonzalo Garcia-Donato and Anabel Forte

Maintainer: <anabel.forte@uv.es>

**References**


See Also

plotBvs for several plots of the result.
pBvs for a parallelized version of Bvs.
gibbsBvs for a heuristic approximation based on Gibbs sampling (recommended when p>20, no other possibilities when p>31).

Examples

```r
## Not run:
#Analysis of Crime Data
#load data
data(UScrime)

#Default arguments are Robust prior for the regression parameters
#and constant prior over the model space
#Here we keep the 1000 most probable models a posteriori:
crime.Bvs<- Bvs(formula="y~.", data=UScrime, n.keep=1000)

#A look at the results:
crime.Bvs

summary(crime.Bvs)

#A plot with the posterior probabilities of the dimension of the
#true model:
plotBvs(crime.Bvs, option="dimension")

#Two image plots of the conditional inclusion probabilities:
plotBvs(crime.Bvs, option="conditional")
plotBvs(crime.Bvs, option="not")

## End(Not run)
```

---

**GibbsBvs**

Bayesian Variable Selection for linear regression models using Gibbs sampling.
Description

Approximate computation of summaries of the posterior distribution using a Gibbs sampling algorithm to explore the model space and frequency of "visits" to construct the estimates.

Usage

GibbsBvs(formula, fixed.cov = c("Intercept"), data, prior.betas = "Robust", prior.models = "Constant", n.iter=1000, init.model = "Full", n.burnin = 50, n.thin=max(1, floor(n.iter / 1000)), time.test = TRUE, priorprobs=NULL)

Arguments

- **formula**: Formula defining the most complex regression model in the analysis. See details.
- **fixed.cov**: A character vector with the names of the covariates that will be considered as fixed (no variable selection over these). This argument provides an implicit definition of the simplest model considered. Default is "Intercept". Use "Null" if selection should be performed over all the variables defined by formula.
- **data**: Data frame containing the data.
- **prior.betas**: Prior distribution for regression parameters within each model. Possible choices include "Robust", "Liangetal", "gZellner", "ZellnerSiow" and "FLS" (see details).
- **prior.models**: Prior distribution over the model space. Possible choices are "Constant", "ScotBerger" and "User" (see details).
- **n.iter**: The total number of iterations performed after the burn in process.
- **init.model**: The model at which the simulation process starts. Options include "Null" (the model only with the covariates specified in fixed.cov), "Full" (the model defined by formula), "Random" (a randomly selected model) and a vector with p (the number of covariates to select from) zeros and ones defining a model.
- **n.burnin**: Length of burn in, i.e. number of iterations to discard at the beginning.
- **n.thin**: Thinning rate. Must be a positive integer. Set 'n.thin' > 1 to save memory and computation time if 'n.iter' is large. Default is 'max(1, floor(n.iter / 1000))'. This parameter jointly with n.iter sets the number of simulations kept and used to construct the estimates.
- **time.test**: If TRUE and the number of variables is large (>=21) a preliminary test to estimate computational time is performed.
- **priorprobs**: A p+1 dimensional vector defining the prior probabilities Pr(M_i) (should be used in the case where prior.models="User"; see the details in Bvs.)

Details

This is a heuristic approximation to the function Bvs so the details there apply also here. The algorithm implemented is a Gibbs sampling-based searching algorithm originally proposed by George and McCulloch (1997). Garcia-Donato and Martinez-Beneito (2013) have shown that this simple sampling strategy in combination with estimates based on frequency of visits (the one here implemented) provides very reliable results.
Value

GibbsBvs returns an object of class bvs with the following elements:

time: The internal time consumed in solving the problem.

lmfull: The lm class object that results when the model defined by `formula` is fitted by lm.

lmnull: The lm class object that results when the model defined by `fixed.cov` is fitted by lm.

variables: The name of all the potential explanatory variables.

n: Number of observations.

p: Number of explanatory variables to select from.

k: Number of fixed variables.

HPMbin: The binary expression of the most probable model found.

inclprob: A data.frame with the estimates of the inclusion probabilities of all the variables.

jointinclprob: A data.frame with the estimates of the joint inclusion probabilities of all the variables.

postprobdim: Estimates of posterior probabilities of the dimension of the true model.

modelslogBF: A matrix with both the binary representation of the visited models after the burning period and the Bayes factor (log scale) of that model to the null model.

call: The call to the function.

method: gibbs

Author(s)

Gonzalo Garcia-Donato and Anabel Forte

References


See Also

plotbvs for a plot of the object created.
Examples

```r
## Not run:
# Analysis of Ozone35 data

data(Ozone35)

# We use here the (Zellner) g-prior for regression parameters and constant prior over the model space.
# In this Gibbs sampling scheme, we perform 10100 iterations, of which the first 100 are discharged (burnin) and of the remaining only one each 10 is kept.
# as initial model we use the full model.
Oz35.GibbsBvs <- GibbsBvs(formula = "y~." , data = Ozone35, prior.betas = "gZellner", prior.models = "Constant", n.iter = 10000, init.model = "Full", n.burnin = 100, time.test = FALSE)

# Note: this is a heuristic approach and results are estimates of the exact answer.

# with the print we can see which is the most probable model among the visited
Oz35.GibbsBvs

# The estimation of inclusion probabilities and the model-averaged estimation of parameters:
summary(Oz35.GibbsBvs)

# Plots:
plotBvs(Oz35.GibbsBvs, option = "conditional")

## End(Not run)
```

### Hald

**Hald data**

The following data relates to an engineering application that was interested in the effect of the cement composition on heat evolved during hardening (for more details, see Woods et al., 1932).

**Usage**

```r
data(Hald)
```

**Format**

A data frame with 13 observations on the following 5 variables.

- `y`  Heat evolved per gram of cement (in calories)
x1 Amount of tricalcium aluminate
x2 Amount of tricalcium silicate
x3 Amount of tetracalcium alumino ferrite
x4 Amount of dicalcium silicate

References

Examples
data(Hald)

---

Ozone35

Ozone35 dataset

Description
Pollution data

Usage
data(Ozone35)

Format
A data frame with 178 observations on the following 36 variables.

y Response = Daily maximum 1-hour-average ozone reading (ppm) at Upland, CA
x4 500-millibar pressure height (m) measured at Vandenberg AFB
x5 Wind speed (mph) at Los Angeles International Airport (LAX)
x6 Humidity (percentage) at LAX
x7 Temperature (Fahrenheit degrees) measured at Sandburg, CA
x8 Inversion base height (feet) at LAX
x9 Pressure gradient (mm Hg) from LAX to Daggett, CA
x10 Visibility (miles) measured at LAX

x4.x4 = x4*x4
x4.x5 = x4*x5
x4.x6 = x4*x6
x4.x7 = x4*x7
x4.x8 = x4*x8
x4.x9 = x4*x9
\[ x_4 \times x_{10} = x_4^* x_{10} \]
\[ x_5 \times x_5 = x_5^* x_5 \]
\[ x_5 \times x_6 = x_5^* x_6 \]
\[ x_5 \times x_7 = x_5^* x_7 \]
\[ x_5 \times x_8 = x_5^* x_8 \]
\[ x_5 \times x_9 = x_5^* x_9 \]
\[ x_5 \times x_{10} = x_5^* x_{10} \]
\[ x_6 \times x_6 = x_6^* x_6 \]
\[ x_6 \times x_7 = x_6^* x_7 \]
\[ x_6 \times x_8 = x_6^* x_8 \]
\[ x_6 \times x_9 = x_6^* x_9 \]
\[ x_6 \times x_{10} = x_6^* x_{10} \]
\[ x_7 \times x_7 = x_7^* x_7 \]
\[ x_7 \times x_8 = x_7^* x_8 \]
\[ x_7 \times x_9 = x_7^* x_9 \]
\[ x_7 \times x_{10} = x_7^* x_{10} \]
\[ x_8 \times x_8 = x_8^* x_8 \]
\[ x_8 \times x_9 = x_8^* x_9 \]
\[ x_8 \times x_{10} = x_8^* x_{10} \]
\[ x_9 \times x_9 = x_9^* x_9 \]
\[ x_9 \times x_{10} = x_9^* x_{10} \]
\[ x_{10} \times x_{10} = x_{10}^* x_{10} \]

Details

This dataset has been used by Garcia-Donato and Martinez-Beneito (2013) to illustrate the potential of the Gibbs sampling method (in BayesVarSel implemented in \texttt{GibbsBvs}).

This data were previously used by Casella and Moreno (2006) and Berger and Molina (2005) and concern \( N = 178 \) measures of ozone concentration in the atmosphere. Of the 10 main effects originally considered, we only make use of those with an atmospheric meaning \( x_4 \) to \( x_{10} \), as was done by Liang et al. (2008). We then have 7 main effects which, jointly with the quadratic terms and second order interactions, produce the above-mentioned \( p = 35 \) possible regressors.

References


**PBvs**

Bayesian Variable Selection for linear regression models using parallel computing.

**Examples**

```r
data(Ozone35)
```

**Description**

PBvs is a parallelized version of Bvs.

**Usage**

```r
PBvs(formula, fixed.cov=c("Intercept"), data, prior.betas = "Robust", prior.models = "Constant", n.keep = 10, n.nodes = 2, priorprobs=NULL, time.test=TRUE)
```

**Arguments**

- **formula**: Formula defining the most complex regression model in the analysis. See details.
- **fixed.cov**: A character vector with the names of the covariates that will be considered as fixed (no variable selection over these). This argument provides an implicit definition of the simplest model considered. Default is "Intercept". Use "Null" if selection should be performed over all the variables defined by formula.
- **data**: data frame containing the data.
- **prior.betas**: Prior distribution for regression parameters within each model. Possible choices include "Robust", "Liangetal", "gZellner", "ZellnerSiow" and "FLS" (see details).
- **prior.models**: Prior distribution over the model space. Possible choices are "Constant" and "ScottBerger" and "User" (see details).
- **n.keep**: How many of the most probable models are to be kept?
- **n.nodes**: Number of nodes to be used in the computation.
- **priorprobs**: A p+1 dimensional vector defining the prior probabilities Pr(M_i) (should be used in the case where prior.models="User"; see details).
- **time.test**: If TRUE a preliminary test to estimate computational time is performed.

**Details**

This function takes advantage of the library **parallel** to distribute the models in the model space throughout the number of nodes available. Its intended use is for moderately large model spaces (p>=20).

A detailed description of the arguments can be found in the details section in Bvs.
Value

PBvs returns an object of class Bvs with the following elements:

- `time` The internal time consumed in solving the problem
- `lmfull` The `lm` class object that results when the model defined by `formula` is fitted by `lm`
- `lmnull` The `lm` class object that results when the model defined by `fixed.cov` is fitted by `lm`
- `variables` The name of all the potential explanatory variables (the set of variables to select from).
- `n` Number of observations
- `p` Number of explanatory variables to select from
- `k` Number of fixed variables
- `HPMbin` The binary expression of the Highest Posterior Probability model
- `modelsprob` A data.frame which summaries the `n.keep` most probable, a posteriori models, and their associated probability.
- `inclprob` A data.frame with the inclusion probabilities of all the variables.
- `jointinclprob` A data.frame with the joint inclusion probabilities of all the variables.
- `postprobdim` Posterior probabilities of the dimension of the true model
- `call` The call to the function
- `method` parallel

Author(s)

Gonzalo Garcia-Donato and Anabel Forte

Maintainer: <anabel.forte@uv.es>

References


See Also

plotBvs for different descriptive plots of the results.
GibbsBvs which implements a heuristic approximation to the problem based on Gibbs sampling.

Examples

## Not run:
#Analysis of Crime Data
#load data
data(UScrime)

#Default arguments are Robust prior for the regression parameters
#and constant prior over the model space
#Here we keep the 1000 most probable models a posteriori:
#The computation over the model space is distributed over two
#cores:
crime.Bvs <- PBvs(formula = "y ~ ", data = UScrime, n.keep = 1000,
n.nodes = 2)

#A look at the results:
crime.Bvs

summary(crime.Bvs)

#An image plot with the joint inclusion
#probabilities:
plotBvs(crime.Bvs, option = "joint")
## End(Not run)

---

plotBvs

*A function for plotting summaries of an object of class Bvs*

Description

Four different plots to summarize graphically the results in an object of class Bvs.

Usage

`plotBvs(x, option = "dimension")`

Arguments

- `x`: An object of class Bvs
- `option`: One of "dimension", "joint", "conditional" or "not"
Details

If option="dimension" this function returns a barplot of the posterior distribution of the dimension of the true model. If option="joint" an image plot of the joint inclusion probabilities is returned. If option="conditional" an image plot of the conditional inclusion probabilities. These should be read as the probability that the variable in the column is part of the true model if the corresponding variables on the row is. Finally, if option="not" the image plot that is returned is that of the the probability that the variable in the column is part of the true model if the corresponding variables on the row is not.

Author(s)

Gonzalo Garcia-Donato and Anabel Forte

Maintainer: <anabel.forte@uv.es>

See Also

See Bvs, PBvs and GibbsBvs for creating objects of the class Bvs.

Examples

```r
#Analysis of Crime Data
data(UScrime)

#Default arguments are Robust prior for the regression parameters and constant prior over the model space
#Here we keep the 1000 most probable models a posteriori:
crime.Bvs<- Bvs(formula="y~.", data=UScrime, n.keep=1000)

#A look at the results:
crime.Bvs

summary(crime.Bvs)

#A plot with the posterior probabilities of the dimension of the true model:
plotBvs(crime.Bvs, option="dimension")

#An image plot of the joint inclusion probabilities:
plotBvs(crime.Bvs, option="joint")

#Two image plots of the conditional inclusion probabilities:
plotBvs(crime.Bvs, option="conditional")
plotBvs(crime.Bvs, option="not")
```
Index

*Topic datasets
  Hald, 12
  Ozone35, 13
*Topic package
  BayesFactor, 3
  BayesVarSel-package, 2
  Bvs, 6
  GibbsBvs, 9
  PBvs, 15

BayesFactor, 3, 3
BayesVarSel (BayesVarSel-package), 2
BayesVarSel-package, 2
Bvs, 3, 5, 6, 10, 15, 18
GibbsBvs, 3, 9, 9, 14, 17, 18

Hald, 12

Ozone35, 13

PBvs, 3, 9, 15, 18
plotBvs, 9, 11, 17, 17
print (Bvs), 6
print.BayesFactor (BayesFactor), 3

summary (Bvs), 6