Package 'R2GUESS'

January 26, 2016

Version 1.7

Date 2016-01-26

Title Wrapper Functions for GUESS

Depends R (>= 3.0.0), fields, MCMCpack, methods, mixOmics, mvtnorm, snowfall

Description Wrapper functions for GUESS, a GPU-enabled sparse Bayesian variable selection method for linear regression based analysis of possibly multivariate/correlated outcomes.

License GPL (>= 2)

SystemRequirements GNU Scientific Library (>= 1.12)

NeedsCompilation yes

Author Gianluca Campanella [cre],
Benoit Liquet [aut],
Marc Chadeau-Hyam [aut],
Leonardo Bottolo [ctb],
Sylvia Richardson [ctb]

Maintainer Gianluca Campanella <gianluca@campanella.org>

Repository CRAN

Date/Publication 2016-01-26 15:19:28

R topics documented:

`R2GUESS-package` *Sparse Bayesian variable selection method for linear regression based analysis of possibly multivariate outcomes.*

Description

R2GUESS is an R package that runs the GUESS code, a computationally optimised C++ implementation of a fully Bayesian variable selection approach (Evolutionary Stochastic Search; ESS algorithm) that can analyse single and multiple responses in an integrated way and can scale to genome-wide datasets. The multi-SNP model underlying GUESS seeks for the best combinations of SNPs to predict the (possibly multivariate) outcome(s). The program uses packages from the GNU Scientific Library (GSL) and offers the possibility to re-route computationally intensive linear algebra operations towards the Graphical Processing Unit (GPU) through the use of proprietary CULA-dense library. The use of GPU-based numerical libraries implies extensive data transfer between the memory/CPU and the GPU, which, in turn, can be computationally expensive. Consequently, for smaller data sets (as the example provided in the package) in which matrix operations are not rate-limiting, the CPU version of GUESS may be more computationally efficient. To ensure an optimal use of the algorithm, and to enable running GUESS on non-CULA compatible systems, the call to GPU-based calculations within GUESS can be switched off through a single argument, and will automatically be disabled in non CULA-compatible systems. Extensive documentation of the source C++ code is available at `http://www.bgx.org.uk/software/GUESS_Doc_short.pdf`.

The current manual details most features of the GUESS algorithm and focuses on the built-in scripts enabling easy runs and automatic post-processing of outputs from GUESS.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

References

Bottolo L, Chadeau-Hyam M ; et al. GUESS-ing polygenic associations with multiple phenotypes using a GPU-based Evolutionary Stochastic Search algorithm. PLoS Genetics. 2013;9(8):e1003657.

Bottolo L, Chadeau-Hyam M ; et al. ESS++ : a C++ Object-Oriented Algorithm for Bayesian Stochastic Search Model Exploration. Bioinformatics. 2011 ; 27 :587-588.

Bottolo L and Richardson S (2010). Evolutionary Stochastic Search for Bayesian model exploration. Bayesian Analysis 5(3), 583-618.

Petretto E, Bottolo L, Langley SR, Heinig M, McDermott-Roe C, Sarwar R, Pravenec M, Hubner N, Aitman TJ, Cook SA and Richardson S (2010). New insights into the genetic control of gene expression using a Bayesian multi-tissue approach. PLoS Comput. Biol., 6(4), e1000737.

See Also

R2GUESS, as.ESS.object, plotMPPI, plot.ESS, print.ESS, summary.ESS

Analysis.permutation *Computing the FDR-controlled level for the significance of the MPPI*

Description

Reads outputs from a preliminary set of permutation runs of GUESS mimicking the null hypothesis of no association (shuffling the responses). Results will subsequently be used to derive the cut-off values for the Marginal Posterior Probability Inclusion of (MPPI) ensuring an empirical FDR control at a user-defined level. Several cut-off values will be investigated and, for each, the corresponding empirical FDR will be returned. The cut-off value providing the closest FDR estimate to the desired level will be retained. Exact match to the desired level will be achieved by linear interpolation. For flexibility, `Analysis.permutation` only performs the calculation of the FDR-controlled cut-off value based on a preliminary set of permuted runs of GUESS, while the function `FDR.permutation` - which is based on the same procedure - also includes the permutations.

Usage

```
Analysis.permutation(x, Npermut, namePermut, threshold = 0.05,
path.output,  number.cutoff=200)
```

Arguments

x	an object of class "ESS".
Npermut	number of permutation runs to base the FDR calculations on. If the number exceeds the number of actual permutations then the calculation will stop and the function will return an error.
namePermut	name specifying the file name stem locating where results from the preliminary permutation runs were stored.
threshold	numeric value specifying the desired the FDR level.
path.output	path directory containing the output files of the permuted runs.
number.cutoff	numeric value specifying the number of points on which to be base the FDR estimation.

Value

Analysis.permutation returns a list containing the following fields:

cutoff.MPI	The MPPI threshold to control empirical FDR at a specified level.
cutoff_int	The linearly interpolation (across the number.cutoff points) cut-off value of MPPI exactly controlling the FDR at the specified level.
cutoff_St	the vector of cutoff values investigated (containing number.cutoff elements).
FDR_emp	Empirical FDR corresponding to the cutoff.MPI value.
FDR_emp_int	Empirical FDR value estimated by linear interpolation for the cutoff_int value.
FDR_emp_St	A vector of empirical FDR values computed by linear interpolation for each cutoff_St values investigated.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

See Also

FDR.permutation

Examples

```
## Not run:
path.output.perm <- tempdir()
path.input.perm <-path.output.perm

cutoff.1 <- FDR.permutation(x=modelY_Hopx,Npermut=6,start.counter=1,
          path.output=path.output.perm,path.input=path.input.perm,nbcpu=3)

cutoff.2 <- FDR.permutation(x=modelY_Hopx,Npermut=6,start.counter=7,
          path.output=path.output.perm,path.input=path.input.perm,nbcpu=3)
```

```
namePermut <- "Permut-Example-GUESS-Y-Hopx"

cutoff.pool <-Analysis.permutation(modelY_Hopx,namePermut=namePermut,
      Npermut=9,threshold=0.05,path.output=path.output.perm,number.cutoff=10)

## End(Not run)
```

`as.ESS.object` *Compiles the main input and output files from a previous run of* R2GUESS *and creates an ESS object.*

Description

The `as.ESS.object` function compiles the main information relating to a previous run of R2GUESS and compiles them into an ESS object to be further analyzed. Main parameters (e.g. `nsweep, burn.in`) are read from the '`feature`' file automatically generated at the end of every R2GUESS run. Main outputs are also included in the object to enable post-processing and further analyses.

Usage

```
as.ESS.object(dataY, dataX, path.input, path.output,
 root.file.output, label.X = NULL, label.Y = NULL, path.par,
 path.init = NULL, file.par, file.init = NULL, file.log = NULL,
 MAP.file = NULL, command=TRUE)
```

Arguments

`dataY`	a character vector (such as '`dataY.txt`') specifying, assuming that data are in the `path.input` folder, the location of the response matrix. In the corresponding file observations are presented in rows, and the (possibly multivariate) outcome(s) in columns. The first two rows (single integers) represent the number of rows (`n`) and columns (`q`) in the matrix.
`dataX`	is a character vector (such as '`dataX.txt`') specifying, assuming that data are in the `path.input` folder, the location of the predictor matrix. In the corresponding file observations are presented in rows, and the predictors in columns. The first two rows (single integers) represent the number of rows (`n`) and columns (`p`) in the matrix.
`path.input`	path linking to the directory containing the data (`dataX` and `dataY`). If `dataX` or/and `dataY` have been entered as data frame(s), the function will generate the corresponding text files required to run GUESS in the `path.input` folder.
`path.output`	path indicating the directory in which output files are saved.
`root.file.output`	name specifying the file stem of the different output files in the `path.output` directory.

label.X — a character vector specifying the name of the predictors. If not specified (=NULL), the variables are labelled by their position in the matrix. Predictors name and information can be given in the MAP.file in the case of SNP data (field SNPName).

label.Y — a character vector specifying the name of the outcomes. If not specified (=NULL), the outcomes are labelled by Y1,..Yq where q is the dimension of the response matrix, or will be the name of the argument dataY (if specified by a data frame).

path.par — path to the directory containing the parameter file (argument file.par)

path.init — path to the directory containing the init file (argument file.init) specifying which variable were included at the first iteration of the MCMC run. By default (file.init=NULL) no init file is required.

file.par — name of the parameter file containing all the user-specified parameters used to set up the run and the features of the moves. This file is located in path.par and contains fields that are extensively described in http://www.bgx.org.uk/software/GUESS_Doc_short.pdf.

file.init — name of the file specifying which variable have been included at the iteration of the MCMC run.

file.log — name of the log file. This file compiles in real time summary information describing the initial parameters, the computational time and state of the run. This file will also contains information about moves sampled at each sweep. By default (=NULL), the name is given by the argument root.file.output extended by '_log' and for computational efficiency (especially when GPU is enabled) a minimal amount of information is returned.

MAP.file — is either a one element character vector or a data frame. If a character vector is used, it specifies, assuming that data are in the path.input folder, the location of the annotation file. In the corresponding file each predictor is presented in rows, and are described as a MAP.file. If a data frame argument is passed, it links to a px3 matrix.

command — Boolean specifying whether the automatically generated C++ command line is saved in the object or not.

Value

An object of class ESS which compiles the following information:

dataY — a character vector defining the location of the response matrix, assuming that data are in the path.input folder.

dataX — a character vector defining the location of the predictor matrix, assuming that data are in the path.input folder.

path.input — path linking to the directory containing the data (dataX and dataY).

path.output — path indicating the directory in which output files were saved.

path.par — path indicating the directory in which to find the parameter file used for the run.

path.init — path indicating the location of file describing the initial guess of the MCMC procedure. If no init files were specified, the field is set to NULL.

time — Boolean value indicating if a file recording the time each sweep took has been created and saved in path.output directory.

file.par	name of the parameter file containing all the user-specified parameters used to set up the run and the features of the moves.
file.init	name of the file specifying which variables were arbitrarily included at the iteration of the MCMC run. If no init file was specified (=NULL), initial guesses were defined by a stepwise regression approach.
file.log	location of the log file.
root.file.output	file name specifying the file stem used to write the output files in the directory specified by path.output.
nsweep	integer specifying the number of sweeps of the MCMC run (including the burn-in).
top	the number of top models that are reported in the output.
BestModels	A list describing the best model visited, with respect to the fields listed in the summary.ESS.
label.X	a character vector specifying the name of the predictors. If not specified (=NULL), the variables are labelled by their position in the matrix from 1 to p.
label.Y	a character vector specifying the name of the outcomes. If not specified (=NULL), the outcomes are labelled by Y1,..Yq, where q is the dimension of the outcome matrix.
p	the number of predictors in the X matrix.
q	the number of outcomes in the response matrix.
n	the number of observations.
nb.chain	the number of chains in the evolutionary algorithm.
burn.in	integer specifying the number of sweeps which were discarded to account for burn-in.
conf	a character vector defining the location of the file compiling observed values for the confounders of interest.
cuda	a boolean value indicating if linear algebra operations have been re-routed towards the GPU.
Egam	a priori average model size.
Sgam	a priori standard deviation of the model size.
MAP.file	a character vector specifying the location of the predictor annotation file, assuming that data are in path.input.
command	a character vector describing the C++ command line used to generate the results, if saved.
seed	the random seed used to initialise the pseudo-random number generator.
Finish	a Boolean value indicating if the run terminated, or was interrupted before reaching the user-defined time limit.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

Examples

```
dataX <- "data-X-C-CODE.txt"
dataY <- "data-Y-ALL-C-CODE.txt"

path.input <- system.file("Input", package="R2GUESS")
path.output <- system.file("Output", package="R2GUESS")
path.par <- system.file("extdata", package="R2GUESS")
file.par <- "Par_file_example_Hopx.xml"
root.file.output <- "Example-GUESS-Y-Hopx"
label.Y <- c("ADR","Fat","Heart","Kidney")
my.env <- new.env()
data(MAP.file,envir=my.env)
MAP.file <- my.env$MAP.file
modelY_Hopx <-as.ESS.object(dataY=dataY,dataX=dataX,path.input=path.input,
    path.output=path.output,root.file.output=root.file.output,label.X=NULL,
    label.Y=label.Y,path.par=path.par,file.par=file.par,MAP.file=MAP.file)

print(modelY_Hopx)
class(modelY_Hopx)
```

`boxplotbeta` *Draws boxplots of the posterior distribution of regression coefficient(s) for a given predictor*

Description

The `boxplot.beta` function draws a boxplot representation of the posterior distribution of regression coefficient(s) for a user-defined predictor. The function requires a preliminary run of `sample.beta`.

Usage

```
boxplotbeta(x, beta, variable)
```

Arguments

`x`	an object of class `ESS`
`beta`	a list produced by `sample.beta` representing the regression coefficient of the predictor of interest with respect to the outcome(s) in the response matrix.
`variable`	name of the predictor of interest, or its position in `dataX`.

Value

Returns a matrix containing samples (iterations in rows) of the regression coefficients (`q` columns) corresponding to the variable specified by the argument `variable` with respect to all outcomes.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

See Also

sample.beta

---

check.convergence *Diagnostic plots for the evaluation of the convergence of the algorithm*

Description

The check.convergence function provides two plots (selectable by which) to investigate the convergence of the posterior distribution towards the target distribution at different stages of the algorithm.

Usage

```
check.convergence(x, which = c(1L:2L), nsplit = 10, nbloc = 4,
             ask = prod(par("mfcol")) < length(which) && dev.interactive())
```

Arguments

x	an object of class ESS.
which	if a subset of the plots is required, specify a subset of the numbers '1:2'.
nsplit	number of splits: the number of sweep intervals to consider.
nbloc	number of moving windows to plot.
ask	ask: logical; if 'TRUE', the user is _ask_ed before each plot, see 'par(ask=.)'

Value

The function check.convergence produces plots to assess the stability of the density estimates of the log posterior distribution. Two sets of graphs are plotted: the first set displays the density estimates of the log posterior distribution based on (i) all the sweep (ii) first half of the sweeps (iii) second half of the sweeps. The second plot represents similar density estimates at different stages of the MCMC procedure according to a sliding window whose characteristics are defined by the number of splits nsplit (i.e. the number of sweep intervals to consider) and number of blocks nbloc (i.e. the number of density estimates to plot).

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

Examples

```
# load an ESS object
modelY_Hopx <- example.as.ESS.object()
check.convergence(modelY_Hopx,nsplit=10,nbloc=4)
check.convergence(modelY_Hopx,nsplit=5,nbloc=4)
check.convergence(modelY_Hopx,nsplit=5,nbloc=2)
```

`data.X` *Data set compiling genotype data for 29 rats.*

Description

This data set contains the genotype data (770 SNPs) for 29 rats.

Format

A `29 x 770` data frame with 29 observations (in rows) and 770 SNPs (in columns)

`data.Y.Hopx` *Data set compiling gene expression levels (HOPX gene) for 29 rats.*

Description

This data set contains gene expression levels for gene Hopx from 4 tissues (Adrenal gland, Fat, Heart, Kidney) in 29 rats.

Format

A `29 x 4` data frame containing the gene expression levels for 29 rats (in rows) in 4 different tissues (in columns): ADR, FAT, Heart, Kidney

---

`example.as.ESS.object` *Function creating an* `ESS` *object from the example files contained in the package*

Description

This functions returns an `ESS` object constructed from the example files provided within the package.

Usage

```
example.as.ESS.object()
```

Value

This functions returns an ESS object constructed from the example files provided within the package. The object contains fields as listed in as.ESS.object. This function is used throughout the documentation to generate the example ESS object used to illustrate the functions embedded the R2GUESS package.

See Also

as.ESS.object

Extend.R2GUESS *Extends an already finished* R2GUESS *run for an extra user-defined number of iterations*

Description

The Extend.R2GUESS extends an already finished R2GUESS run.

Usage

```
Extend.R2GUESS(x, niter, time.limit=NULL)
```

Arguments

x	an object of class ESS.
niter	integer specifying the number of additional sweeps to run the model for.
time.limit	numeric representing the maximum computation time (in hours) allowed for extending the run. By default (NULL), the run will continue until completion.

Details

The Extend.R2GUESS extends an already finished R2GUESS run for an additional niter sweeps. Results from these extra iterations will be appended to the existing history files, and posterior calculations (MPPI, MPP, and the list of best models) will be entirely updated. These additional iterations will use the same pseudo-random number generator as the one used for the initial run, and will initialise it at the state it was while finishing the original run. If the previous run was interrupted before reaching the user-defined time.limit, Extend.R2GUESS will return an error message.

Value

An object of class ESS whose fields are detailed in as.ESS.object, and which contains results from the additional sweeps.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

See Also

Resume.R2GUESS, Postprocess.R2GUESS, as.ESS.object

Examples

```
## Not run:
modelY_Hopx <- example.as.ESS.object()
## Be careful the Output files will be created in modelY_Hopx$path.output
## If you want to save the new Ouput copy/paste the
## folder modelY_Hopx$out in an appropriate place
## and change modelY_Hopx$path.output
modelY_Hopx_extend <- Extend.R2GUESS(modelY_Hopx,100)

## End(Not run)
```

FDR.permutation *Performs parallel permuted runs of* R2GUESS *and returns the empirical FDR-controlled level for the significance of the MPPI*

Description

FDR.permutation will first run GUESS in parallel (over CPUs) for several permutations of the Y matrix mimicking the null hypothesis of no association. Results from the permutation procedure will subsequently be used to derive the cut-off values for the Marginal Posterior Probability of Inclusion (MPPI) ensuring an empirical FDR control at a user-defined level. Several cut-off values will be investigated and, for each, the corresponding empirical FDR will be returned. The cut-off value providing the closest FDR estimate to the desired level will be retained. Exact match to the desired level will be achieved by linear interpolation. The latter is based on the same calculation as in Analysis.permutation.

Usage

```
FDR.permutation(x,path.input = NULL, Npermut, start.counter=1,
   path.output = NULL, threshold = 0.05, nbcpu = NULL, number.cutoff=200)
```

Arguments

x	an object of class ESS.
path.input	path to the directory containing the permuted re-samples of the Y matrix. By default (=NULL), the permuted outcomes are stored in the same directory as the results from the original R2GUESS run (path.input).

path.output	path to the directory in which results from permuted data are stored. By default (=NULL), these are saved in the directory where results from the original GUESS run were stored (path.output).
Npermut	number of permutations to run.
start.counter	defines the integer from which to start labelling permutation runs.
threshold	numeric value specifying the desired FDR level.
nbcpu	integer indicating the number of CPUs to use for the permutation procedure. This number has to be lower than the number of cores available on the platform. By default (=NULL), the function uses a single core.
number.cutoff	numeric value specifying the number of points on which to base the FDR estimation.

Value

FDR.permutation generates permutation re-samples from the original Y matrix and generates for each permutation standard R2GUESS output files. Sets of results can be separately analysed using Analysis.permutation. The function also returns a list containing the following fields:

cutoff.MPI	the MPPI threshold to control empirical FDR at a specified level.
cutoff_int	the linearly interpolated (across the number.cutoff points) cut-off value of MPPI exactly controlling the FDR at the specified level.
cutoff_St	the vector of cut-off values investigated (containing number.cutoff elements).
FDR_emp	empirical FDR corresponding to the cutoff.MPI value.
FDR_emp_int	empirical FDR value estimated by linear interpolation for the cutoff_int value.
FDR_emp_St	a vector of empirical FDR values computed by linear interpolation for each cutoff_St values investigated.

Author(s)

Benoit Liquet <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

See Also

Analysis.permutation

Examples

```
## Not run:
modelY_Hopx <- example.as.ESS.object()
path.output.perm <- tempdir()
path.input.perm <-path.output.perm

cutoff <- FDR.permutation(x=modelY_Hopx,Npermut=100,start.counter=1,
           path.output=path.output.perm,path.input=path.input.perm,nbcpu=3)

## End(Not run)
```

`get.g.sweep` *Internal function used to generate the regression coefficients. This function extracts the values of the shrinkage factor g for a given model specified by its ranked posterior probability*

Description

Internal function used for the generation of the regression coefficients beta. `get.g.sweep` extracts the values of the shrinkage factor `g` associated with the sweeps at which a given model has been visited. The model under investigation is defined by its rank in terms of posterior probability.

Usage

```
get.g.sweep(x, res, model)
```

Arguments

`x`	an object of class `ESS`.
`res`	a matrix of boolean values indicating at which sweeps of the MCMC run the model has been visited. This matrix is the output of `get.sweep.best.model`.
`model`	integer specifying the rank (according to the model posterior probability) of the model under investigation.

Value

An object of class `g.prior` containing:

`g.value`	the observed values of `g` for the chosen model.
`predictors`	the variables belonging to the model considered.
`X`	the X design matrix for the model.
`Y`	the centred outcome matrix.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

See Also

`get.sweep.best.model`

`get.sweep.best.model` *Internal function used to generate the regression coefficients. This function extracts the sweep(s) for which each selected models has been visited along the MCMC run.*

Description

Internal function used to generate the regression coefficients. This function extracts the sweep(s) at which each of the top models has been visited along the MCMC run. The rank of the model(s) to look up for is a parameter of the function.

Usage

```
get.sweep.best.model(x, models = 1)
```

Arguments

`x`	an object of class `ESS`.
`models`	a vector containing the rank of the models to look up for.

Value

A list containing a Boolean matrix (`N.model` rows and `x$nsweep` columns), where the element `i,j` indicates if the `i-th` model (ranked according to its decreasing posterior probability) has been visited at sweep `j`.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

`MAP.file` *MAP file describing genotypes from the rats experiment.*

Description

This data set contains the annotation information for the 770 SNPs used in the example. SNPs are described in rows (in the same order as they appear in `data.X`), and the following information is provided: the SNP rs name (`SNPName`), the Chromosome it belongs to (`Chr`), and the physical position on the chromosome (`Posn`)

Format

A data frame with 770 rows and 3 columns

See Also

data.X

---

pairwise.correlation *Calculates and plots the pairwise correlation between outcomes*

Description

The pairwise.correlation function plots an image with the pairwise correlation between phenotypes and provides the corresponding source matrix.

Usage

```
pairwise.correlation(Y, label.Y = NULL)
```

Arguments

Y	data frame or matrix corresponding to the responses.
label.Y	character vector indicating the label of the outcomes. By default (label.Y=NULL) labels of Y are set to colnames(Y).

Value

A matrix of the pairwise correlation between outcomes.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

Examples

```
data(data.Y.Hopx)
   res.cor.Y.Hopx <-
   pairwise.correlation(data.Y.Hopx,label.Y=NULL)$matcor
```

plot.ESS *Provides diagnostic plots to assess the convergence of the MCMC procedure along the run*

Description

The plot.ESS function provides a set of four plots summarising the behaviour of all chains along the Evolutionary Monte Carlo run. These plots are used to assess the convergence of the algorithm and comprise: (i)- the history plot of the shrinkage factor g, (ii)- the history plot of the model size in each chain, (iii)- the history plot of the log posterior in each chain, and (iv)- the history plot of each chain's temperature (during the burn-in only, as temperatures are fixed at the end of the burn-in).

Usage

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

Arguments

x	an object of class ESS.
...	additional arguments: range.SF range value for the shrinkage factor plot (g); range.MS range value for the model size plot; range.LP range value for the log posterior plot.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

Examples

```
# load an ESS object
modelY_Hopx <- example.as.ESS.object()
plot(modelY_Hopx)
```

plotcim *Clustered Image Maps (CIMs) (heat maps)*

Description

This function generates color-coded Clustered Image Maps (CIMs) representing the pairwise correlation between a subset of predictors and the outcomes.

Usage

```
plotcim(x, select.variable, labelX = NULL)
```

Arguments

x	an object of class ESS.
select.variable	name of the selected predictors.
labelX	names of the predictors used when running R2GUESS. This argument is optional if a MAP.file has been provided.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

Examples

```
modelY_Hopx <- example.as.ESS.object()
MPPI.Hopx <-
plotMPPI(modelY_Hopx,threshold.model=20,threshold.variable=0.25,Figure=FALSE)
plotcim(modelY_Hopx,select.variable=MPPI.Hopx$var.TOP.MPI)
```

---

plotcim.explore *Plots a cluster image mapping of correlations between outcomes and all predictors*

Description

The plot.cim.explore function plots a cluster image mapping of correlations between outcomes and all the predictors.

Usage

```
plotcim.explore(matX, matY)
```

Arguments

matX	data frame corresponding to the predictors.
matY	data frame corresponding to the outcomes.

Details

To be used with a small number of predictors (<1,000).

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

Examples

```
data(data.Y.Hopx)
data(data.X)
plotcim.explore(data.X,data.Y.Hopx)
```

---

plotmodel *Visualisation of the proximity between best models*

Description

The plotmodel function provides an image plot of the proximity (according to an overlap metric) between the best models.

Usage

```
plotmodel(x, threshold.model = 20)
```

Arguments

x	an object of class ESS.
threshold.model	either an integer representing the number of models to be retained in the list of best models, or a value defining the minimal model posterior probability for inclusion.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

Examples

```
# load an ESS object
modelY_Hopx <- example.as.ESS.object()
plotmodel(modelY_Hopx,20)
```

plotMPPI *Plots the marginal posterior probability of inclusion (MPPI) for each predictor*

Description

The plotMPPI function plots the marginal posterior probability of inclusion (MPPI) of each predictor.

Usage

```
plotMPPI(x, threshold.model = 0.01,
  threshold.variable = 0.1, Figure = TRUE, cutoff = TRUE, useMC = FALSE)
```

Arguments

x	an object of class ESS.
threshold.model	either an integer representing the number of model to be retained in the list of best models, or a value defining the minimal model posterior probability for inclusion.
threshold.variable	threshold probability for selecting the most relevant predictors. This threshold can be calibrated by controlling the FDR using FDR.permutation.
Figure	if TRUE (by default) will generate the MPPI plot. If FALSE only information on the selected predictors will provided.
cutoff	if TRUE (by default) will plot an horizontal line representing the cut-off value indicating by the argument threshold.variable. If FALSE the cut-off value is not plotted.
useMC	if TRUE, use simple Monte Carlo estimation for the MPPI across all visited models.

Value

The plotMPPI function returns information on the best models (i.e. those satisfying the threshold.model criterion) and on the most relevant predictors. (above threshold.variable).

Rank	the rank on the models selected.
nVisits	number of times each model has been visited along the run.
ModSize	number of predictors in each of the best models.
logCondPost	the log conditional posterior for each model.
Jeffries	Jeffrie's scale value for each model.
postProb	posterior probability of each model.
modelName	list of predictors in each of the best models.

modelPosInX	position (in the predictor matrix) of the constituents of the best models.
var.TOP.MPI	predictors with MPPI>threshold.variable and belonging to the best models.
var.MPI	predictors which have a MPPI greater than threshold.variable.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

Examples

```
modelY_Hopx <- example.as.ESS.object()
# To get a large plot
# dev.new(width=13,height=6)
MPPI.Hopx <- plotMPPI(modelY_Hopx,threshold.model=20,threshold.variable=0.45)
print(MPPI.Hopx)
```

---

plotvariable *Visualisation of the best models*

Description

The plotvariable function provides a plot indicating the variables included in each of the best models.

Usage

```
plotvariable(x, threshold.model = 0.05, file.annotation = NULL)
```

Arguments

x	an object of class ESS.
threshold.model	either an integer representing the number of model to be retained in the list of best models, or a value defining the minimal model posterior probability.
file.annotation	text file containing the name of the predictors. This file is optional if a MAP.file has been specified.

Value

A list of predictors included in the best models.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

Examples

```
# load an ESS object
modelY_Hopx <- example.as.ESS.object()
plotvariable(modelY_Hopx,20)
```

Postprocess.R2GUESS *Performs posterior inference from an interrupted* R2GUESS *run.*

Description

The Postprocess.R2GUESS function calculates the MPPI, the MPP and lists the best models visited based on a previous run which has been interrupted due to computing time exceeding the user-defined limit.

Usage

```
Postprocess.R2GUESS(x)
```

Arguments

x	an object of class ESS.

Details

The Postprocess.R2GUESS function calculates the MPPI, the MPP and lists the best models visited based on a previous run which has been interrupted due to computing time exceeding the user-defined limit. This function is used to explore results from a run which has been interrupted and provides intermediate posterior inference for the MPPI, MPP, and list of best visited models.

Value

An object of class ESS whose fields are detailed in as.ESS.object.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

See Also

Resume.R2GUESS, Extend.R2GUESS, as.ESS.object

Examples

```
## Not run:
## First we are creating a run which has been not finished in 1 hour
path.input <- system.file("Input", package="R2GUESS")
path.output <- tempdir()
path.par <- system.file("extdata", package="R2GUESS")
file.par.Hopx <- "Par_file_example_Hopx.xml"
print(paste(path.par,file.par.Hopx,sep=""))
root.file.output.Hopx <- "Example-GUESS-Y-Hopx"
label.Y <- c("ADR","Fat","Heart","Kidney")
data(data.Y.Hopx)
data(data.X)
data(MAP.file)

modelY_Hopx<-R2GUESS(dataY=data.Y.Hopx,dataX=data.X,,choice.Y=1:4,
label.Y=label.Y,MAP.file=MAP.file,file.par=file.par.Hopx,
file.init=NULL,file.log=NULL,root.file.output=root.file.output.Hopx,
path.input=path.input,path.output=path.output,path.par=path.par,
path.init=NULL,nsweep=510000,burn.in=10000,Egam=5,Sgam=5,top=100,
history=TRUE,time=TRUE,nb.chain=3,conf=0,cuda=FALSE,time.limit=1)

modelY_Hopx_postprocess <- Postprocess.R2GUESS(modelY_Hopx)


## End(Not run)
```

`print.ESS` *Provides a* `print` *method for class* `ESS`

Description

Print method for R2GUESS

Usage

```
## S3 method for class 'ESS'
print(x, ...)
```

Arguments

x	an object of class ESS
...	not used currently

Details

print method for ESS class, returns a description of the x object.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

See Also

as.ESS.object

Examples

```
# load an ESS object
modelY_Hopx <- example.as.ESS.object()
print(modelY_Hopx)
```

R2GUESS *Wrapper function that reads the input files and parameter values required by GUESS, runs the C++ code from R and stores the main GUESS output in an* ESS *object*

Description

The R2GUESS function reads and compiles data, input files and parameters that are required to run GUESS source code. It automatically runs GUESS (enabling or not the GPU capacity), saves the results and summary files in text files. For portability, R2GUESS generates an ESS object which compiles information about the input and parameters used to run GUESS, and outputs as detailed in as.ESS.object.

Usage

```
R2GUESS(dataY, dataX, path.input, path.output, path.par,
  path.init = NULL, file.par, file.init = NULL,
  file.log = NULL, nsweep, burn.in, Egam,
  Sgam, root.file.output, time = TRUE, top = 100,
  history = TRUE, label.X = NULL, label.Y = NULL,
  choice.Y = NULL, nb.chain, conf = NULL, cuda = TRUE,
  MAP.file = NULL, time.limit=NULL,seed=NULL)
```

Arguments

dataY	either a one element character vector (such as 'dataY.txt') or a data frame. If dataY is entered as a character vector, it specifies, assuming that data are in the path.input folder, the location of the response matrix. In the corresponding file observations are presented in rows, and the (possibly multivariate) outcome(s) in columns. The first two rows (single integers) represent the number of rows (n) and columns (q) in the matrix. If a data frame argument is passed, it links to a nxq numerical matrix compiling the observed responses.
dataX	either a one element character vector (such as 'dataX.txt') or a data frame. If dataX is entered as a character vector, it specifies, assuming that data are in the path.input folder, the location of the predictor matrix. In the corresponding file observations are presented in rows, and the predictors in columns. The first two rows (single integers) represent the number of rows (n) and columns (p) in the matrix. If a data frame argument is passed, it links to a nxq numerical matrix compiling the observed predictors.
path.input	path linking to the directory containing the data (dataX and dataY). If dataX or/and dataY have been entered as data frame(s), the function will generate the corresponding text files required to run GUESS in the path.input folder.
path.output	path indicating the directory in which output files will be saved.
path.par	path indicating the directory in which to find the parameter file needed to run GUESS.
path.init	path indicating the location of the file describing the initial guess of the MCMC procedure (i.e. the variables to include in the initial model).
file.par	name of the parameter file containing all user-specified parameters required to set up the run and the features of the moves. This file is located in path.par and contains fields that are extensively described in http://www.bgx.org.uk/software/GUESS_Doc_short.pdf. These parameters are not mandatory and, if not specified, they will be set to their default values, also given in documentation. An example of this file is provided in the package.
file.init	name of the file specifying which variables to include at the first iteration of the MCMC run. The first row of the file is a single scalar representing the number of rows (# variables to include). Subsequent rows indicate the position of the covariates to include. This file is optional and if not specified (default=NULL), initial guesses of the MCMC algorithm will be derived from a step-wise regression approach.
file.log	name of the log file. This file compiles in real time summary information describing the initial parameters, the computational time and state of the run. This file will also contain information about moves sampled at each sweep. By default (=NULL), the name is given by the argument root.file.output extended by '_log' and for computational efficiency (especially when GPU is enabled), a minimal amount of information is returned.
nsweep	integer specifying the number of sweeps for the MCMC run (including the burn-in).
burn.in	integer specifying the number of sweeps to be discarded to account the burn-in.
Egam	numeric representing the 'a priori' average model size.

Sgam	numeric representing the 'a priori' standard deviation of the model size.
root.file.output	name specifying the file stem for writing the output files in the directory specified by path.output.
time	Boolean value. When time=TRUE (default value) a file recording the time each sweep took will be created and saved in path.output directory.
top	number of top models to be reported in the output. The default value is 100.
history	Boolean value. When history=TRUE (default value), a number of additional output files that record the history of each move is provided. See section 5 of http://www.bgx.org.uk/software/GUESS_Doc_short.pdf for more details.
label.X	a character vector specifying the name of the predictors. If not specified (=NULL), variables are labelled by their position in the matrix. Predictors name and information is given in the MAP.file in the case of SNP data (field SNPName).
label.Y	a character vector specifying the name of the outcomes. If not specified (=NULL), the outcomes are labelled Y1,..Yq, where q is the number of columns in the outcome matrix or will be named after the argument dataY (if specified by a data frame).
choice.Y	a character vector or a numeric vector specifying which phenotypes in the response matrix dataY to analyse in a joint model. By default, all phenotypes in the response matrix will be considered.
nb.chain	an integer specifying the number of chains to consider in the evolutionary procedure.
conf	either a one element character vector (such as 'conf.txt') or a data frame. If conf is entered as a character vector, it specifies, assuming that data are in the path.input folder, the location of the confounder matrix. In the corresponding file observations are presented in rows, and the values for the confounders in columns. The first two rows (single integers) represent the number of rows (n) and columns (k) in the matrix. If a data frame argument is passed, it links to a nxk numerical matrix compiling the observed confounders. If specified, the function will substitute the response matrix by the residuals from the linear model regressing the confounders against the outcomes.
cuda	a boolean value. cuda=TRUE redirects linear algebra operations towards the GPU. On non-CULA compatible platforms, this option will be ignored.
MAP.file	either a one element character vector or a data frame. If a character vector is used, it specifies, assuming that data are in the path.input folder, the location of the annotation file. In the corresponding file, predictors are presented in rows, and are described as a MAP.file. If a data frame argument is passed, it links to a px3 matrix.
time.limit	a numerical value specifying the maximum computing time (in hours) for the run. If the run exceeds that value, modelling options, parameters value, state of the pseudo random number generator, and state of each chain will be saved to enable to resume the run exactly at the same point it was interrupted (using resume option). By default (=NULL) the run will go on until its completion.
seed	a integer specifying the random seed used to initialize the pseudo-random number generator. If not specified, the seed will be initialised using the CPU clock.

Details

For any of the dataX, dataY parameters, if a data frame argument is passed, a text file labelled data-*-C-CODE.txt will be created in the path.input directory. If conf is specified, and additional files representing the adjusted responses will be created according to the file labelling system.This file will be formatted to have the suitable structure to be read by the C++ code: individuals presented in rows, and observations in columns, with the first two rows indicating the number of rows and columns in the matrix. The returned ESS object will include all result files produced by the code. The number and type of outputs produced depend on the running options chosen. A full description of the available output can be found in http://www.bgx.org.uk/software/GUESS_Doc_short.pdf

Value

An object of class ESS containing information listed in as.ESS.object. The object can subsequently be used to post-process the results using provided R functions (such as summary.ESS, plotMPPI, plot.ESS).

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

See Also

as.ESS.object, summary.ESS,as.ESS.object, plotMPPI, plot.ESS

Examples

```
## Not run:
path.input <- system.file("Input", package="R2GUESS")
path.output <- tempdir()
path.par <- system.file("extdata", package="R2GUESS")
file.par.Hopx <- "Par_file_example_Hopx.xml"
#you can have a look of the parameter file in
print(paste(path.par,file.par.Hopx,sep=""))
##To reach convergence you may need to increase nsweep=110000 and the burn.in=10000
## RUNNING is APPROX 5 minutes
root.file.output.Hopx <- "Example-GUESS-Y-Hopx"
label.Y <- c("ADR","Fat","Heart","Kidney")
data(data.Y.Hopx)
data(data.X)
data(MAP.file)

modelY_Hopx<-R2GUESS(dataY=data.Y.Hopx,dataX=data.X,choice.Y=1:4,
label.Y=label.Y,,MAP.file=MAP.file,file.par=file.par.Hopx,file.init=NULL,
file.log=NULL,root.file.output=root.file.output.Hopx,path.input=path.input,
path.output=path.output,path.par=path.par,path.init=NULL,nsweep=11000,
burn.in=1000,Egam=5,Sgam=5,top=100,history=TRUE,time=TRUE,
nb.chain=3,conf=NULL,cuda=FALSE)
```

```
summary(modelY_Hopx,20) # 20 best models

print(modelY_Hopx)

## End(Not run)
```

Resume.R2GUESS *Function resuming an interrupted* R2GUESS *run*

Description

The Resume.R2GUESS function will resume a run that was interrupted due to computation time exceeding the user-defined time limit.

Usage

```
Resume.R2GUESS(x, time.limit=NULL)
```

Arguments

x	an ESS object corresponding to a preliminary R2GUESS run which was interrupted due to computing time exceeding the user-specified time limit.
time.limit	a numerical value specifying the maximum computing time (in hours) for the continuation of the run. If the run exceeds that value, modelling options, parameters value, state of the pseudo random number generator, and state of each chain will be saved to enable to resume the run exactly at the same point it was interrupted (using resume option). By default (=NULL) the resuming run will go on until its completion.

Details

The Resume.R2GUESS function will continue a run that was interrupted due to computation time exceeding the user-defined time limit. It will compile data, input files and parameters that are required for running GUESS code. Additional iterations required to complete the run will be appended to the existing history files, and posterior calculations (MPPI, MPP, and the list of best models) will be added to the ESS object. The function will use the same pseudo-random number generator as the one used for the initial run, and will initialise it at the state it was at the end of the original run.

Value

If the resuming run completes the run, it will return the complete ESS object (compiling information listed in as.ESS.object function).

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

See Also

Extend.R2GUESS, Postprocess.R2GUESS, as.ESS.object

Examples

```
## Not run:
## First we are creating a run which has been not finished in 1 hour
path.input <- system.file("Input", package="R2GUESS")
path.output <- tempdir()
path.par <- system.file("extdata", package="R2GUESS")
file.par.Hopx <- "Par_file_example_Hopx.xml"
print(paste(path.par,file.par.Hopx,sep=""))
root.file.output.Hopx <- "Example-GUESS-Y-Hopx"
label.Y <- c("ADR","Fat","Heart","Kidney")
data(data.Y.Hopx)
data(data.X)
data(MAP.file)

modelY_Hopx<-R2GUESS(dataY=data.Y.Hopx,dataX=data.X,choice.Y=1:4,
label.Y=label.Y,,MAP.file=MAP.file,file.par=file.par.Hopx,
file.init=NULL,file.log=NULL,root.file.output=root.file.output.Hopx,
path.input=path.input,path.output=path.output,path.par=path.par
,path.init=NULL,nsweep=510000,burn.in=10000,Egam=5,Sgam=5,top=100
,history=TRUE,time=TRUE,nb.chain=3,conf=0,cuda=FALSE,time.limit=1)

modelY_Hopx_resume <- Resume.R2GUESS(modelY_Hopx)


## End(Not run)
```

---

sample.beta *Posterior distribution of the regression coefficients for a chosen model*

Description

The sample.beta function generates the effect size estimates of a chosen model within the best models.

Usage

```
sample.beta(x, res.g, Nmonte.sigma = 1, Nmonte = 1)
```

Arguments

x	an object of class ESS
res.g	an object of class g.prior produced by get.g.sweep.
Nmonte.sigma	number of re-samples of the posterior variance covariance matrix of the outcomes (Sigma), for a given value of g among those observed for the model under investigation.
Nmonte	number of re-samples of the regression coefficient vector, for a given value of g and of the Sigma matrix.

Value

A list containing the sampled values of the regression coefficients. Re-samples for a given value of g among those observed for the model under investigation are presented in rows (Nmonte x Nmonte.sigma rows) and columns are arranged such that the k-th block of q values represents the regression coefficients of predictor k for all q outcomes.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

Examples

```
modelY_Hopx <- example.as.ESS.object()
n.sweep <- get.sweep.best.model(modelY_Hopx,models=1:2)
res.g <- get.g.sweep(modelY_Hopx,n.sweep$result,model=1)
beta <- sample.beta(modelY_Hopx,res.g,Nmonte=5,Nmonte.sigma=5)
res.D14Mit3 <- boxplotbeta(modelY_Hopx,beta,variable="D14Mit3")
```

---

summary.ESS summary *method for class* ESS

Description

Summary method for ESS objects.

Usage

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

Arguments

object	an object of class ESS.
...	integer indicating the number of best models to be summarized.

Details

summary method for the ESS class. It returns a summary of the x object including the list of the best models visited

Value

The function summary returns a data frame containing the following fields describing all of the best models according to:

Rank	the rank of the model according to its posterior probability.
nVisits	the number of times the model has been visited during the run (including the burn-in).
FirstVisit	the iteration at which the model was first visited.
nEvalBefore1st	the number of iteration before the first visit of the model.
ModeSize	the number of variables in the model.
logCondPost	the log posterior probability (integrated over the shrinkage factor g).
postProb	the posterior probability of the model.
jeffrey	the Jeffrey's scale for the model.
modelName	the list of variables (predictors) included in the model.

Author(s)

Benoit Liquet, <b.liquet@uq.edu.au>,
Marc Chadeau-Hyam <m.chadeau@imperial.ac.uk>,
Leonardo Bottolo <l.bottolo@imperial.ac.uk>,
Gianluca Campanella <g.campanella11@imperial.ac.uk>

Examples

```
# load an ESS object
modelY_Hopx <- example.as.ESS.object()
summary(modelY_Hopx,20)
```

Index