Package 'PReMiuM'

May 1, 2014

Type Package

Title Dirichlet Process Bayesian Clustering, Profile Regression

Version 3.0.26

Date 2014-04-29

Author David I. Hastie <david.hastie@rsimony.com>, Silvia Liverani <liveranis@gmail.com>, Sylvia Richardson <sylvia.richardson@mrc-bsu.cam.ac.uk>, Lamiae Azizi <lamiae.azizi@mrc-bsu.cam.ac.uk>

Maintainer Silvia Liverani <liveranis@gmail.com>

Description Dirichlet process Bayesian clustering, also known as profile regression.

URL http://www.silvialiverani.com/PReMiuM/

License GPL-2

LazyLoad yes

Depends R (>= 3.0.2)

Imports Rcpp (>= 0.11), ggplot2 (>= 0.9.2.1), cluster, plotrix (>= 3.5)

LinkingTo Rcpp, RcppEigen (>= 0.3), BH (>= 1.54)

SystemRequirements GNU make

R topics documented:

PReMiuM-package   *Dirichlet Process Bayesian Clustering*

Description

Dirichlet process Bayesian clustering and functions for the post-processing of its output.

Details

Package:	PReMiuM
Type:	Package
Version:	3.0.24
Date:	2014-04-04
License:	GPL3
LazyLoad:	yes

Program to implement Dirichlet Process Bayesian Clustering as described in Liverani et al. 2013. Previously this project was called profile regression.

Details

PReMiuM provides the following:

- Implements an infinite Dirichlet process model
- Can do dependent or independent slice sampling (Kalli et al., 2011) or truncated Dirichlet process model (Ishwaran and James, 2001)
- Handles categorical or Normal covariates, or a mixture of them
- Handles Bernoulli, Binomial, Categorical, Poisson or Normal responses
- Handles inclusion of fixed effects in the response model
- Handles Extra Variation in the response (for Bernoulli, Binomial and Poisson response only)
- Handles variable selection (tested in Discrete covariate case only)

- Includes label switching moves for better mixing
- Allows user to exclude the response from the model
- Allows user to compute the entropy of the allocation
- Allows user to run with a fixed alpha or update alpha (default)
- Allows users to run predictive scenarios (at C++ run time)
- Basic or Rao-Blackwellised predictions can be produced
- Handling of missing data
- C++ for model fitting
- Uses Eigen Linear Algebra Library and Boost C++
- Completely self contained (all library code in included in distribution)
- Adaptive MCMC where appropriate
- R package for generating simulation data and post processing
- R plotting functions allow user choice of what to order clusters by

Authors

David Hastie, Department of Epidemiology and Biostatistics, Imperial College London, UK

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

Acknowledgements

References

Molitor J, Papathomas M, Jerrett M and Richardson S. (2010) Bayesian Profile Regression with an Application to the National Survey of Children's Health, Biostatistics 11: 484-498.

Papathomas M, Molitor J, Richardson S. et al (2011) Examining the joint effect of multiple risk factors using exposure risk profiles: lung cancer in non smokers. Environmental Health Perspectives 119: 84-91.

Hastie, D. I., Liverani, S., Azizi, L., Richardson, S. and Stucker I. (2013) A semi-parametric approach to estimate risk functions associated with multi-dimensional exposure profiles: application to smoking and lung cancer. *BMC Medical Research Methodology*. To appear.

Molitor, J., Brown, I. J., Papathomas, M., Molitor, N., Liverani, S., Chan, Q., Richardson, S., Van Horn, L., Daviglus, M. L., Stamler, J. and Elliott, P. (2013) Blood pressure differences associated with DASH-like lower sodium compared with typical American higher sodium nutrient profile: INTERMAP USA. *Submitted*.

Hastie, D. I., Liverani, S. and Richardson, S. (2014) Sampling from Dirichlet process mixture models with unknown concentration parameter: Mixing issues in large data implementations. *Submitted*. Available at http://uk.arxiv.org/abs/1304.1778

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2014) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted*. Available at http://uk.arxiv.org/abs/1303.2836

Examples

```
## Not run:
# example for Poisson outcome and Discrete covariates
inputs <- generateSampleDataFile(clusSummaryPoissonDiscrete())
runInfoObj<-profRegr(yModel=inputs$yModel,
    xModel=inputs$xModel, nSweeps=10, nClusInit=20,
    nBurn=20, data=inputs$inputData, output="output",
    covNames = inputs$covNames, outcomeT = inputs$outcomeT,
    fixedEffectsNames = inputs$fixedEffectNames)

dissimObj<-calcDissimilarityMatrix(runInfoObj)
clusObj<-calcOptimalClustering(dissimObj)
riskProfileObj<-calcAvgRiskAndProfile(clusObj)
clusterOrderObj<-plotRiskProfile(riskProfileObj,"summary.png")

## End(Not run)
```

---

calcAvgRiskAndProfile   *Calculation of the average risks and profiles*

Description

Calculation of the average risks and profiles.

Usage

```
calcAvgRiskAndProfile(clusObj, includeFixedEffects=F)
```

Arguments

clusObj   Object of type clusObj.

includeFixedEffects

> By default this is set to FALSE. If it is set to FALSE then the risk profile is
> computed with the parameters beta of the fixed effects assumed equal to zero. If
> it is set to TRUE, then risk profile at each sweep is computed adjusting for the
> sample of the beta parameter at that sweep.

Value

A list with the following components. This is an object of type riskProfileObj.

riskProfClusObj

> The object of type clusObj as given in the input of this function.

risk   A matrix that has a column for each cluster and a row for each sweep. Each el-
                   ement of the matrix represents the estimated risk at each sweep for each cluster.

profile	An array whose first dimension is the number of sweeps, the second is the number of clusters, the third is the number of discrete covariates and the fourth is the number of categories of each of the covariates. Each element of the array represents the covariate profile at each sweep for each cluster. The fourth dimension does not exists if the covariate type is Normal. If the covariate type is mixed, then instead of this element, the two elements below are defined, 'profilePhi' and 'profileMu'.
profileStar	This is NULL if there has not been any variable selection. otherwise it contains the
empiricals	A vector of length of the optimal number of clusters, where each value is the empirical mean of the outcome for each cluster.
profileStdDev	An array whose first dimension is the number of sweeps, the second is the number of clusters, the third and the fourth are the number of continuous covariates. Each square matrix identified by the first and second dimension of the array represents the standard deviation at each sweep for each cluster. This element is only available if the covariate type is continuous or mixed.
profilePhi	This array is the equivalent of the 'profile' above for discrete covariates in case of mixed covariates.
profileStarPhi	This array is defined as profile and profilePhi, but the values are computed only if a variable selection procedure has been run. The definition of the star profile is given in Liverani, S., Hastie, D. I. and Richardson, S. (2013) PReMiuM: An R package for Bayesian profile regression.
profileMu	This array is the equivalent of the 'profile' above for Normal covariates in case of mixed covariates.
profileStarMu	This array is defined as profile and profileMu, but the values are computed only if a variable selection procedure has been run. The definition of the star profile is given in Liverani, S., Hastie, D. I. and Richardson, S. (2013) PReMiuM: An R package for Bayesian profile regression.

Authors

David Hastie, Department of Epidemiology and Biostatistics, Imperial College London, UK

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted*. Available at http://uk.arxiv.org/abs/1303.2836

Examples

```
## Not run:
generateDataList <- clusSummaryBernoulliDiscrete()
inputs <- generateSampleDataFile(generateDataList)
```

```
runInfoObj<-profRegr(yModel=inputs$yModel, xModel=inputs$xModel, nSweeps=10,
    nBurn=20, data=inputs$inputData, output="output", nClusInit=15,
    covNames=inputs$covNames)

dissimObj<-calcDissimilarityMatrix(runInfoObj)
clusObj<-calcOptimalClustering(dissimObj)
riskProfileObj<-calcAvgRiskAndProfile(clusObj)

## End(Not run)
```

---

calcDissimilarityMatrix

Calculates the dissimilarity matrix

Description

Calculates the dissimilarity matrix.

Usage

```
calcDissimilarityMatrix(runInfoObj, onlyLS=FALSE)
```

Arguments

runInfoObj   Object of type runInfoObj.

onlyLS   Logical. It is set to FALSE by default. When it is equal to TRUE the dissimilar-
                ity matrix is not returned and the only method available to identify the optimal
                partition using 'calcOptimalClustering' is least squares. This parameter is to be
                used for datasets with many subjects, as C++ can compute the dissimilarity ma-
                trix but it cannot pass it to R for usage in the function 'calcOptimalClustering'.
                As guidance, be aware that a dataset with 85,000 subjects will require a RAM
                of about 26Gb, even if onlyLS=TRUE.

Value

Need to write this

disSimRunInfoObj
                These are details regarding the run and in the same format as runInfoObj.

disSimMat   The dissimilarity matrix, in vector format. Note that it is diagonal, so this con-
                tains the upper triangle diagonal entries.

disSimMatPred   The dissimilarity matrix, again in vector format as above, for the predicted sub-
                jects.

lsOptSweep   The optimal partition among those explored by the MCMC, as defined by the
                least squares method. See Dahl (2006).

onlyLS   Logical. If it set to TRUE the only method available to identify the optimal
                partition using 'calcOptimalClustering' is least squares.

Authors

David Hastie, Department of Epidemiology and Biostatistics, Imperial College London, UK

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted.* Available at http://uk.arxiv.org/abs/1303.2836

Examples

```
generateDataList <- clusSummaryBernoulliDiscrete()
inputs <- generateSampleDataFile(generateDataList)
runInfoObj<-profRegr(yModel=inputs$yModel, xModel=inputs$xModel,
    nSweeps=10, nBurn=20, data=inputs$inputData, output="output",
    covNames=inputs$covNames,nClusInit=15)

dissimObj<-calcDissimilarityMatrix(runInfoObj)
```

---

calcOptimalClustering   *Calculation of the optimal clustering*

Description

Calculates the optimal clustering.

Usage

```
calcOptimalClustering(disSimObj, maxNClusters=NULL, useLS=F)
```

Arguments

disSimObj	A dissimilarity matrix (in vector format, as the output of the function calcDissimilarityMatrix(), and as described in ?calcDissimilarityMatrix) or a list of dissimilarity matrix, to combine the output of several runs of the MCMC.
maxNClusters	Set the maximum number of clusters allowed. This is set to the maximum number explored.
useLS	This is set to FALSE by default. If it is set to TRUE then the least-squares method is used for the calculation of the optimal clustering, as described in Molitor et al (2010). Note that this is set to TRUE by default if disSimObj$onlyLS is set to TRUE.

Value

the output is a list with the following elements. This is an object of type clusObj.

clusObjRunInfoObj
> Details on this run. An object of type runInfoObj.

clusterSizes   Cluster sizes.

clusteringPred   The predicted cluster memberships for the predicted scenarios.

clusObjDisSimMat
> Dissimilarity matrix.

clustering   Cluster memberships.

nClusters   Optimal number of clusters.

avgSilhouetteWidth
> Average silhouette width when using medoids method for clustering.

Authors

David Hastie, Department of Epidemiology and Biostatistics, Imperial College London, UK

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted*. Available at http://uk.arxiv.org/abs/1303.2836

Examples

```
## Not run:
generateDataList <- clusSummaryBernoulliDiscrete()
inputs <- generateSampleDataFile(generateDataList)
runInfoObj<-profRegr(yModel=inputs$yModel, xModel=inputs$xModel,
    nSweeps=10, nBurn=20, data=inputs$inputData, output="output",
    covNames=inputs$covNames, nClusInit=15)

dissimObj<-calcDissimilarityMatrix(runInfoObj)
clusObj<-calcOptimalClustering(dissimObj)

## End(Not run)
```

calcPredictions   *Calculates the predictions*

Description

Calculates the predictions.

Usage

```
calcPredictions(riskProfObj, predictResponseFileName=NULL,
    doRaoBlackwell=F, fullSweepPredictions=F, fullSweepLogOR=F)
```

Arguments

riskProfObj   Object of type riskProfObj.

predictResponseFileName

If this function is run after the function profRegr, and outcome (and possibly fixed effects) are known for the predicted profiles, then there is no need to set this, as the function profRegr will have produced a file ending in "_predict-Full.txt". This file allows the computation of measures of fit for cross-validation. If the file has not been produced automatically, it can be produced manually and it can be provided here. We discourage this and we provide no documentation for doing so.

doRaoBlackwell   By default this is set to FALSE. If it is set to TRUE then Rao-Blackwell predictions are computed.

fullSweepPredictions

By default this is set to FALSE. If it is set to TRUE then a prediction is computed for each sweep.

fullSweepLogOR   By default this is set to FALSE. If it is set to TRUE then a prediction log OR is computed for each sweep.

Value

The output is a list with the following elements.

bias   The bias of the predicted values with respect to the observed outcome. If the response is not provided, this is set to NA.

rmse   The root mean square error of the predicted values with respect to the observed outcome. If the response is not provided, this is set to NA.

mae   The mean absolute error of the predicted values with respect to the observed outcome. If the response is not provided, this is set to NA.

observedY   The values of the outcome provided by the user. This is in the case that predictions are run as a validation tool. If the response is not provided, this is set to NA.

predictedY   This matrix has as many rows as predictions requested by the user. It is the
                  median of the predicted values over all the sweeps that have been run after the
                  burn-in period.

doRaoBlackwell   This is set to TRUE if it has done Rao-Blackwell predictions, and FALSE oth-
                  erwise.

predictedYPerSweep

                  This array has the first dimension equivalent to the number of sweeps and the
                  second dimension as large as the number of predictions requested by the user. It
                  contains the predicted values per sweep.

logORPerSweep   This array has the first dimension equivalent to the number of sweeps and the
                  second dimension as large as the number of predictions requested by the user. It
                  contains the predicted log OR values per sweep (not available for Poisson and
                  Normal outcome).

Details

This functions computes predicted responses, for various prediction scenarios. It is assumed that
the predictive allocations and Rao-Blackwell predictions have already been done in profRegr using
the 'predict' input.

The user can provide the function profRegr with a data.frame through the predict argument. This
data.frame has a row for each subject, where each row contains values for the response, fixed effects
and offset / number of trials (depending on the response model) where available. Missing values
in this data.frame are denoted by 'NA'. If the data.frame is not provided then the response, fixed
effect and offset data is treated as missing for all subjects. If a subject is missing fixed effect
values, then the mean value or 0 category fixed effect is used in the predictions (i.e. no fixed effect
contribution to predicted response). If the offset / number of trials is missing this value is taken to
be 1 when making predictions. If the response is provided for all subjects, the predicted responses
are compared with the observed responses and the bias and rmse are computed. If the response is
provided in the data frame it must be in a column called "outcome".

The function can produce predicted values based on simple allocations (the default), or a Rao-
Blackwellised estimate of predictions, where the probabilities of allocations are used instead of
actually performing a random allocation.

Authors

David Hastie, Department of Epidemiology and Biostatistics, Imperial College London, UK

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC
Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R
package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted*. Available at
http://uk.arxiv.org/abs/1303.2836

Examples

```
## Not run:
inputs <- generateSampleDataFile(clusSummaryBernoulliDiscrete())

# prediction profiles
preds<-data.frame(matrix(c(0, 0, 1, 0, 0,
0, 0, 1, NA, 0),ncol=5,byrow=TRUE))
colnames(preds)<-names(inputs$inputData)[2:(inputs$nCovariates+1)]

# run profile regression
runInfoObj<-profRegr(yModel=inputs$yModel, xModel=inputs$xModel,
    nSweeps=100, nBurn=1000, data=inputs$inputData, output="output",
    covNames=inputs$covNames,predict=preds)

# postprocessing
dissimObj <- calcDissimilarityMatrix(runInfoObj)
clusObj <- calcOptimalClustering(dissimObj)
riskProfileObj <- calcAvgRiskAndProfile(clusObj)
clusterOrderObj <- plotRiskProfile(riskProfileObj,"summary.png",
    whichCovariates=c(1,2))
output_predictions <- calcPredictions(riskProfileObj,fullSweepPredictions=TRUE)

# example where the fixed effects can be provided for prediction
# but the observed response is missing
# (there are 2 fixed effects in this example).
# in this example we also use the Rao Blackwellised predictions

inputs <- generateSampleDataFile(clusSummaryPoissonNormal())

# prediction profiles
predsPoisson<- data.frame(matrix(c(7, 2.27, -0.66, 1.07, 9,
    -0.01, -0.18, 0.91, 12, -0.09, -1.76, 1.04, 16, 1.55, 1.20, 0.89,
    10, -1.35, 0.79, 0.95),ncol=5,byrow=TRUE))
colnames(predsPoisson)<-names(inputs$inputData)[2:(inputs$nCovariates+1)]

# run profile regression
runInfoObj<-profRegr(yModel=inputs$yModel,
        xModel=inputs$xModel, nSweeps=100,
        nBurn=100, data=inputs$inputData, output="output",
        covNames = inputs$covNames, outcomeT="outcomeT",
        fixedEffectsNames = inputs$fixedEffectNames,predict=predsPoisson)

# postprocessing
dissimObj<-calcDissimilarityMatrix(runInfoObj)
clusObj<-calcOptimalClustering(dissimObj)
riskProfileObj<-calcAvgRiskAndProfile(clusObj)
output_predictions <- calcPredictions(riskProfileObj,fullSweepPredictions=TRUE)


# example where both the observed response and fixed effects are present
#(there are no fixed effects in this example, but
# these would just be added as columns between the first and last columns).
```

```
inputs <- generateSampleDataFile(clusSummaryPoissonNormal())

# prediction profiles
predsPoisson<- data.frame(matrix(c(NA, 2.27, -0.66, 1.07, NA,
      -0.01, -0.18, 0.91, NA, -0.09, -1.76, 1.04, NA, 1.55, 1.20, 0.89,
      NA, -1.35, 0.79, 0.95),ncol=5,byrow=TRUE))
colnames(predsPoisson)<-names(inputs$inputData)[2:(inputs$nCovariates+1)]

# run profile regression
runInfoObj<-profRegr(yModel=inputs$yModel,
         xModel=inputs$xModel, nSweeps=10,
         nBurn=20, data=inputs$inputData, output="output",
         covNames = inputs$covNames, outcomeT="outcomeT",
         fixedEffectsNames = inputs$fixedEffectNames,
         nClusInit=15, predict=predsPoisson)

# postprocessing
dissimObj<-calcDissimilarityMatrix(runInfoObj)
clusObj<-calcOptimalClustering(dissimObj)
riskProfileObj<-calcAvgRiskAndProfile(clusObj)
output_predictions <- calcPredictions(riskProfileObj,fullSweepPredictions=TRUE)


## End(Not run)
```

clusSummaryBernoulliDiscrete

Sample datasets for profile regression

Description

Definition of skeleton of sample datasets for profile regression.

Usage

```
clusSummaryBernoulliDiscrete()
clusSummaryBernoulliNormal
clusSummaryBernoulliDiscreteSmall()
clusSummaryBinomialNormal()
clusSummaryCategoricalDiscrete()
clusSummaryNormalDiscrete()
clusSummaryNormalNormal()
clusSummaryPoissonDiscrete()
clusSummaryPoissonNormal()
clusSummaryVarSelectBernoulliDiscrete()
clusSummaryBernoulliMixed()
```

Value

The output of these function is a list with the following components. These can be used as inputs for profile regression function profRegr().

outcomeType	The outcome type of the dataset.
covariateType	The covariate type of the dataset.
nCovariates	The number of covariates generated.
nCategories	The number of categories of the covariates if the covariates are discrete or mixed.
nFixedEffects	The number of fixed effects.
fixedEffectsCoeffs	
	The names of the fixed effects.
missingDataProb	
	The pobability of generating missing data.
nClusters	The number of clusters.
clusterSizes	The number of observations in each cluster.
clusterData	The dataset, including the outcome, the covariates, the fixed effects, the number of trials (if Binomial outcome) and the offset (for Poisson outcome).
covNames	The names of the covariates of the dataset.
nDiscreteCovs	The number of discrete covariates, if the covariate type is mixed.
nContinuousCovs	
	The number of continuous covariates, if the covariate type is mixed.
outcomeT	The name of the column of the dataset containing the number of trials (if Binomial outcome) or the offset (for Poisson outcome).

Details

clusSummaryBernoulliDiscrete generates a dataset with Bernoulli outcome and discrete covariates.

clusSummaryBernoulliNormal generates a dataset with Bernoulli outcome and Normal covariates.

clusSummaryBernoulliDiscreteSmall generates a dataset with Bernoulli outcome and discrete covariates (with smaller cluster sizes).

clusSummaryBinomialNormal generates a dataset with Binomial outcome and discrete covariates.

clusSummaryCategoricalDiscrete generates a dataset with categorical outcome and discrete covariates.

clusSummaryNormalDiscrete generates a dataset with Normal outcome and discrete covariates.

clusSummaryNormalNormal generates a dataset with Normal outcome and Normal covariates.

clusSummaryPoissonDiscrete generates a dataset with Poisson outcome and discrete covariates.

clusSummaryPoissonNormal generates a dataset with Poisson outcome and Normal covariates.

clusSummaryVarSelectBernoulliDiscrete generates a dataset with Bernoulli outcome and discrete covariates, suitable for variable selection as some covariates are not driving the clustering.

clusSummaryBernoulliMixed generates a dataset with Bernoulli outcome and mixed covariates.

Authors

David Hastie, Department of Epidemiology and Biostatistics, Imperial College London, UK

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted*. Available at http://uk.arxiv.org/abs/1303.2836

Examples

```
clusSummaryBernoulliDiscrete()
```

```
computeRatioOfVariance
```

computeRatioOfVariance

Description

Computes of the ratio between the variance of the extra variation and the total variance.

Usage

```
computeRatioOfVariance(runInfoObj)
```

Arguments

This function can only be used when the extra variation is included in the response model.

Object of type runInfoObj

Value

runInfoObj   For each sweep this function outputs the ratio between the variance of the thetas' and the sum of the variances of the thetas' and the extra variation epsilon as described in Liverani et al. (2013).

Authors

David Hastie, Department of Epidemiology and Biostatistics, Imperial College London, UK

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, Azizi, L., D. I., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Bayesian profile regression. *Submitted*. Available at http://uk.arxiv.org/abs/1303.2836

---

generateSampleDataFile

Generate sample data files for profile regression

Description

Generation of random sample datasets for profile regression.

Usage

```
generateSampleDataFile(clusterSummary)
```

Arguments

clusterSummary   A vector of strings of the covariate names as by the column names in the data argument.

Value

The output of this function is a list with the following elements

yModel	The outcome model according to which the data has been generated.
xModel	The covariate model according to which the data has been generated.
inputData	The data.frame that contains the data.
covNames	The names of the covariates.
fixedEffectNames	
	The names of the fixed effects.

Authors

David Hastie, Department of Epidemiology and Biostatistics, Imperial College London, UK

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted*. Available at http://uk.arxiv.org/abs/1303.2836

Examples

```
# generation of data for clustering

generateDataList <- clusSummaryBernoulliDiscrete()
inputs <- generateSampleDataFile(generateDataList)
```

---

globalParsTrace   *Plot of the trace of some of the global parameters*

Description

Function to lot the trace of some global parameters

Usage

```
globalParsTrace(runInfoObj, parameters = "nClusters",plotBurnIn=FALSE,whichBeta=1)
```

Arguments

	This function allows to visualise the trace of the global parameters.
	Note that this function has not been optimised for large datasets.
	An object of class runInfoObj.
runInfoObj parameters	The parameter whose trace will be plotted. This can be set equal to "nClusters" (default), "alpha", "mpp" and "beta", as by the model. As beta can be a vector, we advise to also set the option "whichBeta" below to select which fixed effect parameter to visualise in the plot. "mpp" stands for marginal partition posterior, also referred to as marginal model posterior.
plotBurnIn	Set to FALSE (default) it does not plot the trace for the burn in period. Set to TRUE it plots the trace including the burn in period.
whichBeta	Integer which selects which fixed effect parameter is plotted.

Value

Plot of trace of some global parameters.

Authors

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted*. Available at http://uk.arxiv.org/abs/1303.2836

Examples

```
# generate simulated dataset
generateDataList <- clusSummaryBernoulliDiscreteSmall()
inputs <- generateSampleDataFile(generateDataList)

# run profile regression
runInfoObj<-profRegr(yModel=inputs$yModel, xModel=inputs$xModel,
 nSweeps=10, nBurn=20, data=inputs$inputData, output="output", nFilter=3,
 covNames=inputs$covNames,nClusInit=15,reportBurnIn=FALSE,
 fixedEffectsNames = inputs$fixedEffectNames)

# plot trace for alpha
globalParsTrace(runInfoObj,parameters="alpha",plotBurnIn=FALSE)
```

---

heatDissMat   *Plot the heatmap of the dissimilarity matrix*

Description

Function to plot the heatmap of the dissimilarity matrix

Usage

```
heatDissMat(dissimObj, main=NULL, xlab=NULL, ylab=NULL)
```

Arguments

dissimObj	An object of class dissimObj.
main	The usual plot option, to be passed to the heatmap function.
ylab	The usual plot option, to be passed to the heatmap function.
xlab	The usual plot option, to be passed to the heatmap function.

Value

Plot of the heatmap of the dissimilary matrix. This functions uses the function 'heatmap' of package 'stats'. Note that this function has not been optimised for large datasets.

Authors

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted.* Available at http://uk.arxiv.org/abs/1303.2836

Examples

```
## Not run:
# generate simulated dataset
generateDataList <- clusSummaryBernoulliDiscreteSmall()
inputs <- generateSampleDataFile(generateDataList)

# run profile regression
runInfoObj<-profRegr(yModel=inputs$yModel, xModel=inputs$xModel,
 nSweeps=10, nBurn=2000, data=inputs$inputData, output="output",
 covNames=inputs$covNames,nClusInit=15)

# compute dissimilarity matrix
dissimObj<-calcDissimilarityMatrix(runInfoObj)

# plot heatmap
heatDissMat(dissimObj)

## End(Not run)
```

---

is.wholenumber   *Function to check if a number is a whole number*

Description

Function to check if a number is whole, accounting for a rounding error.

Usage

```
is.wholenumber(x, tol = .Machine$double.eps^0.5)
```

Arguments

x   The number to be checked.

tol   Tolerance level.

Value

The default method for 'is.wholenumber' returns 'TRUE' if the number provided is a whole number.

Authors

David Hastie, Department of Epidemiology and Biostatistics, Imperial College London, UK

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted*. Available at http://uk.arxiv.org/abs/1303.2836

Examples

```
is.wholenumber(4) # TRUE
is.wholenumber(3.4) # FALSE
```

---

margModelPosterior   *Marginal Model Posterior*

Description

Compute the marginal model posterior.

Usage

```
margModelPosterior(runInfoObj,allocation)
```

Arguments

runInfoObj	An object of type runInfoObj.
allocation	By default, if allocation is not provided, the _z.txt file is read to compute the marginal model posterior for all the partitions available there. If allocation is equal to a vector that corresponds to a partition, the marginal model posterior is computed for that given partition.

Value

It returns a file in the output folder, with name ending in "_margModPost.txt", that contains the marginal model posterior. It also returns the mean of the values of the marginal model posterior as they appear in the file ending in "_margModPost.txt" in the output folder.

Authors

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted.* Available at http://uk.arxiv.org/abs/1303.2836

Examples

```
inputs <- generateSampleDataFile(clusSummaryBernoulliDiscrete())

runInfoObj<-profRegr(yModel=inputs$yModel,
         xModel=inputs$xModel, nSweeps=5,
         nBurn=10, data=inputs$inputData, output="output",
         covNames = inputs$covNames, nClusInit=15,
         fixedEffectsNames = inputs$fixedEffectNames)

margModelPosterior(runInfoObj)
```

---

plotPredictions   *Plot the conditional density using the predicted scenarios*

Description

Plots the conditional density for the predicted scenarios provided. It produces a pdf with a page for each predictive scenario provided. Each page has a plot of the predicted response, in the order as they were provided to the function. Note that fixed effects are not processed in this function.

Usage

```
plotPredictions(outfile, runInfoObj, predictions,
    logOR=FALSE)
```

Arguments

outfile	String. The name of the output PDF file. The default is "condDensity.pdf".
runInfoObj	An object of type runInfoObj which contains all the details about the run of profRegr.
predictions	An object of type predictions which contains all the details about the run of calcPredictions.
logOR	Whether to plot the response probability or log odds ratios. The default is FALSE and the response probability is plotted.

Value

The output is a plot in PDF format.

Authors

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted.* Available at http://uk.arxiv.org/abs/1303.2836

Examples

```
## Not run:
# example with Bernoulli outcome and Discrete covariates
inputs <- generateSampleDataFile(clusSummaryBernoulliDiscrete())
# prediction profiles
preds<-data.frame(matrix(c(
2, 2, 2, 2, 2,
0, 0, NA, 0, 0),ncol=5,byrow=TRUE))

colnames(preds)<-names(inputs$inputData)[2:(inputs$nCovariates+1)]
# run profile regression
runInfoObj<-profRegr(yModel=inputs$yModel, xModel=inputs$xModel,
 nSweeps=10000, nBurn=10000, data=inputs$inputData, output="output",
 covNames=inputs$covNames,predict=preds,
 fixedEffectsNames = inputs$fixedEffectNames)
dissimObj <- calcDissimilarityMatrix(runInfoObj)
clusObj <- calcOptimalClustering(dissimObj)
riskProfileObj <- calcAvgRiskAndProfile(clusObj)
predictions <- calcPredictions(riskProfileObj,fullSweepPredictions=TRUE,fullSweepLogOR=TRUE)

plotPredictions(outfile="predictiveDensity.pdf",runInfoObj=runInfoObj,
 predictions=predictions,logOR=TRUE)


## End(Not run)
```

| plotRiskProfile | *Plot the Risk Profiles* |

Description

Plots the risk profiles for a profile regression model.

Usage

```
plotRiskProfile(riskProfObj, outFile, showRelativeRisk=F,
    orderBy=NULL, whichClusters=NULL,
    whichCovariates=NULL, useProfileStar=F)
```

Arguments

`riskProfObj`	An object of type riskProfObj.
`outFile`	Path and file name to save the plot.
`showRelativeRisk`	
	Whether to show the relative risk (with respect to the risk of the first cluster). This option is not available for Normal outcome.
`orderBy`	Order by which the clusters are to be displayed. It can take values "Empirical", "ClusterSize" and "Risk" (the latter only if the outcome is provided). It can also take the name of a covariate to order the clusters, in which case the clusters are ordered.
`whichClusters`	Either a vector of indeces that corresponds to the clusters that are to be displayed. The length of this vector must be greater than 1. The default is that all clusters are shown.
`whichCovariates`	
	Either a vector of indeces or a vector of strings that corresponds to the covariates that are to be displayed. The length of this vector must be greater than 1. The default is that all covariates are shown.
`useProfileStar`	To be set equal to TRUE only if a variable selection procedure has been run. The definition of the star profile is given in Liverani, S., Hastie, D. I. and Richardson, S. (2013) PReMiuM: An R package for Bayesian profile regression.

Value

This function creates a png plot saved in the path given by outFile. All clusters are visually displayed together.

For discrete covariates, instead of plotting the probability that a phi is above or below the mean value, we plot the actual phi values (and plot the mean value across clusters as a horizontal line).

For normal covariates, for each covariate the upper plot is the posterior distribution for the mean mu, and the lower plot is the posterior distribution of sqrt(Sigma[j,j]) (i.e. the standard deviation for that covariate).

It also returns the following vector.

`meanSortIndex`	This vector is the index that represents the order that the clusters are represented. The default ordering is by empirical risk.

Authors

David Hastie, Department of Epidemiology and Biostatistics, Imperial College London, UK

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted*. Available at http://uk.arxiv.org/abs/1303.2836

Examples

```
## Not run:
# example for Poisson outcome and Discrete covariates
inputs <- generateSampleDataFile(clusSummaryPoissonDiscrete())
runInfoObj<-profRegr(yModel=inputs$yModel,
    xModel=inputs$xModel, nSweeps=10, nClusInit=15,
    nBurn=20, data=inputs$inputData, output="output",
    covNames = inputs$covNames, outcomeT = inputs$outcomeT,
    fixedEffectsNames = inputs$fixedEffectNames)

dissimObj<-calcDissimilarityMatrix(runInfoObj)
clusObj<-calcOptimalClustering(dissimObj)
riskProfileObj<-calcAvgRiskAndProfile(clusObj)
clusterOrderObj<-plotRiskProfile(riskProfileObj,"summary.png")

## End(Not run)
```

---

profRegr	*Profile Regression*

Description

Fit a profile regression model.

Usage

```
profRegr(covNames, fixedEffectsNames, outcome="outcome",
        outcomeT=NA, data, output="output", hyper, predict,
        nSweeps=1000, nBurn=1000, nProgress=500, nFilter=1,
        nClusInit, seed, yModel="Bernoulli", xModel="Discrete",
        sampler="SliceDependent", alpha=-2, dPitmanYor = 0, excludeY=FALSE,
        extraYVar=FALSE, varSelectType="None", entropy,reportBurnIn=FALSE,
        run=TRUE, discreteCovs, continuousCovs, whichLabelSwitch="123")
```

Arguments

covNames
A vector of strings of the covariate names as by the column names in the data argument.

fixedEffectsNames
A vector of strings of the fixed effect names as by the column names in the data argument. Each fixed effect must be of class 'numeric'. If a fixed effect is of class 'character', an error message will appear and the fixed effect will need to be recoded as numeric.

outcome
A string of column of the data argument that contains the outcome. The outcome cannot have missing values - you could consider predicting the value of the outcome for those subjects for which it has not been observed.

outcomeT	A string of column of the data argument that contains the offset (for Poisson outcome) or the number of trials (for Binomial outcome).
data	A data frame which has as columns the outcome, the covariates, the fixed effects if any and the offset (for Poisson outcome) or the number of trials (for Binomial outcome) or censoring (for Survival outcome). The outcome cannot have missing values - you could consider predicting the value of the outcome for those subjects for which it has not been observed. For Survival response censoring must be coded as 0 if the event has not occurred (ie, there has been censoring) and 1 if the event has occurred (no censoring has taken place).
output	Path to folder to save all output files. The covariates can have missing values, which must be coded as 'NA'. There cannot be missing values in the fixed effects - if there are, use an imputation method before using profile regression.
hyper	Object of type setHyperparams with hyperparameters specifications. This is optional, default values are provided for all hyperparameters. See ?setHyperparams for details.
predict	Data frame containing the predictive scenarios. This is only required if predictions are requested.
	At each iteration the predictive subjects are assigned to one of the current clusters according to their covariate profiles (but ignoring missing values), or their Rao Blackwellised estimate of theta is recorded (a weighted average of all theta, weighted by the probability of allocation into each cluster.
	The predictive subjects have no impact on the likelihood and so do not determine the clustering or parameters at each iteration. The predictive allocations are then recorded as extra entries in each row of the output_z.txt file. This can then be processed in the post processing to create a dissimilarity matrix with the fitting subjects. The post procesing function calcPredictions will create predicted response values for these subjects.
	See ?calcPredictions for more details and examples.
nSweeps	Number of iterations of the MCMC after the burn-in period. By default this is 1000.
nBurn	Number of initial iterations of the MCMC to be discarded. By default this is 1000.
reportBurnIn	If TRUE then the burn in iterations are reported in the output files, if set to FALSE they are not. It is set to FALSE by default.
nProgress	The number of sweeps at which to print a progress update. By default this is 500.
nFilter	The frequency (in sweeps) with which to write the output to file. The default value is 1.
nClusInit	The number of clusters individuals should be initially randomly assigned to (Unif[50,60]).
seed	The value for the seed for the random number generator. The default value is the current time.
yModel	The model type for the outcome variable. The options currently available are "Bernoulli", "Poisson", "Binomial", "Categorical", "Normal" and "Survival". The default value is Bernoulli.

xModel	The model type for the covariates. The options currently available are "Discrete", "Normal" and "Mixed". The default value is "Discrete". Note that there are some memory leaks using option "Mixed" so while it is an available option, we advise to use it with care.
sampler	The sampler type to be used. Options are "SliceDependent", "SliceIndependent" and "Truncated". The default value is "SliceDependent".
alpha	The value to be used if alpha is fixed. If a value smaller than or equal to -1 is used then alpha is random, if dPitmanYor is equal to zero (the random alpha option is available for Dirichlet process prior only). The default value is -2 (random alpha). For fixed alpha, if dPitmanYor is in the interval (0,1) then a Pitman-Yor process prior is used instead of a Dirichlet process prior.
dPitmanYor	The discount parameter for the Pitman-Yor process prior. The default value is 0, which is equivalent to a Dirichlet process prior. This parameter must belong to the interval [0,1) and it must be provided together with a non-negative value for alpha. The Pitman-Yor process prior is only available for non-random parameters. Note that the third label switching move is only available for Dirichlet process priors, so it will not be run if dPitmanYor>0. Therefore setting dPitmanYor to a value greater than zero will forse whichLabelSwitch=12.
excludeY	If TRUE only the covariate data X is modelled. By default this is set to FALSE.
extraYVar	If set equal to TRUE extra Gaussian variance is included in the response model. This option is available only for Bernoulli, Binomial and Poisson response. By default the extra Gaussian variance is not included, so extraYVar=FALSE.
varSelectType	The type of variable selection to be used "None", "BinaryCluster" or "Continuous". The "BinaryCluster" variable selection is the implementation of the novel variable selection formulation proposed by Papathomas, Molitor, Hoggart, Hastie, Richardson (2012) "Exploring data from genetic association studies using Bayesian variable selection and the Dirichlet process: application to searching for gene x gene patterns" in Genetic Epidemiology. The "Continuous" variable selection is based on the method proposed by Chung and Dunson (2009) "Nonparametric Bayes conditional distribution modelling with variable selection" in the Journal of the American Statistical Association. Both types of variable selection can be used with discrete, continuous or mixed covariates. The default value is "None".
entropy	If included then we compute allocation entropy. By default the allocation entropy is not included.
run	Logical. If TRUE then the MCMC is run. Set run=FALSE if the MCMC has been run already and it is only required to collect information about the run.
discreteCovs	The names of the discrete covariates among the covariate names, if xModel="Mixed". This and continuousCovs must be defined if xModel="Mixed", while covNames is ignored.
continuousCovs	The names of the discrete covariates among the covariate names, if xModel="Mixed". This and continuousCovs must be defined if xModel="Mixed", while covNames is ignored.
whichLabelSwitch	
	The label switching moves to run. The options available are moves 1, 2 and 3 ("123"), moves 1 and 2 ("12") and move 3 only ("3"). The moves are described

in Hastie et al. (2013). Note that the third label switching move is only available for Dirichlet process priors, so it will not be run if dPitmanYor>0. Therefore setting dPitmanYor to a value greater than zero will forse whichLabelSwitch=12.

Value

Once the C++ has completed the output from fitting the regression is stored in a number of text files in the directory specified. Files are produced containing the MCMC traces for all of the values of interest, along with a log file and files for monitoring the acceptance rates of the adaptive Metropolis Hastings moves.

It returns a number of files in the output directory as well as a list with the following elements. This an object of type runInfoObj.

directoryPath	String. Directory path of the output files.
fileStem	String. The
inputFileName	String. Location and file name of input dataset as created by this function for the C++ routines
nSweeps	Integer. The number of sweeps of the MCMC after the burn-in.
nBurn	Integer. The number of iterations in the burn-in period of the MCMC.
reportBurnIn	Logical. Whether the output of the burn-in report should be included.
nFilter	Integer. The frequency (in sweeps) with which to write the output to file.
nProgress	The number of sweeps at which to print a progress update.
nSubjects	Integer. The number of subjects.
nPredictSubjects	
	Integer. The number of subjects for which to run predictions.
fullPredictFile	
	Logical. It is FALSE by default. It is equal to TRUE if the outcome or the outcome and the fixed effects were included in the dataframe provided in the input predict. If TRUE, the function will have a produced a file ending in "_predictFull.txt" which contains the values of the outcome and fixed effects for the computation of measures of fit in the function calcPredictions.
covNames	A vector of strings with the names of the covariates.
xModel	String. The model type for the covariates.
includeResponse	
	Logical. If FALSE only the covariate data X is modelled.
yModel	String. The model type for the outcome.
varSelect	Logical. If FALSE no variable selection is performed.
varSelectType	String. It specifies what type of variable selection has been performed, if any.
nCovariates	Integer. The number of covariates.
nFixedEffects	Integer. The number of fixed effects.
nCategoriesY	Integer. The number of categories of the outcome, if yModel = "Categorical". It is 1 otherwise.

nCategories	Vector of integers. The number of categories of each covariate, if xModel = "Discrete". It is 1 otherwise.
extraYVar	TRUE if extra Gaussian variance is included in the response model.
xMat	A matrix of the covariate data.
yMat	A matrix of the outcome data, including the offset if the outcome is Poisson, the number of trials if the outcome is Binomial and 0 or 1 for Survival outcome (1 for censored individuals, 0 otherwise).
wMat	A matrix of the fixed effect data.
whichLabelSwitch	The label switching moves that have been run. The options available are moves 1, 2 and 3 ("123"), moves 1 and 2 ("12") and move 3 only ("3"). The moves are described in Hastie et al. (2013).

Authors

David Hastie, Department of Epidemiology and Biostatistics, Imperial College London, UK

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

and a contribution for mixed covariates by Lamiae Azizi, MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted.* Available at http://uk.arxiv.org/abs/1303.2836

Hastie, D. I., Liverani, S. and Richardson, S. (2014) Sampling from Dirichlet process mixture models with unknown concentration parameter: Mixing issues in large data implementations. *Submitted.* Available at http://uk.arxiv.org/abs/1304.1778

Examples

```
# example for Poisson outcome and Discrete covariates
inputs <- generateSampleDataFile(clusSummaryPoissonDiscrete())
runInfoObj<-profRegr(yModel=inputs$yModel,
    xModel=inputs$xModel, nSweeps=10, nClusInit=20,
    nBurn=20, data=inputs$inputData, output="output",
    covNames = inputs$covNames, outcomeT = inputs$outcomeT,
    fixedEffectsNames = inputs$fixedEffectNames)


# example with Bernoulli outcome and Mixed covariates
#inputs <- generateSampleDataFile(clusSummaryBernoulliMixed())
#runInfoObj<-profRegr(yModel=inputs$yModel,
#    xModel=inputs$xModel, nSweeps=10, nClusInit=15,
#    nBurn=20, data=inputs$inputData, output="output",
#    discreteCovs = inputs$discreteCovs,
#    continuousCovs = inputs$continuousCovs)
```

---

setHyperparams	*Definition of characteristics of sample datasets for profile regression*

Description

Hyperparameters for the priors can be specified here and passed as an argument to profRegr.

The user can specify some or all hyperparameters. Those hyperparameters not specified will take their default values. Where the file is not provided, all hyperparameters will take their default values.

Usage

```
setHyperparams(shapeAlpha=NULL,rateAlpha=NULL,
    aPhi=NULL,mu0=NULL,Tau0=NULL,R0=NULL,
    kapp0=NULL,muTheta=NULL,sigmaTheta=NULL,dofTheta=NULL,muBeta=NULL,
    sigmaBeta=NULL,dofBeta=NULL,shapeTauEpsilon=NULL,
    rateTauEpsilon=NULL,aRho=NULL,bRho=NULL,atomRho=NULL,shapeSigmaSqY=NULL,
    scaleSigmaSqY=NULL,rSlice=NULL,truncationEps=NULL)
```

Arguments

shapeAlpha	The shape parameter for Gamma prior on alpha (default=2)
rateAlpha	The inverse-scale (rate) parameter for the Gamma prior on alpha (default=1)
aPhi	The vector of parameters for the Dirichlet prior on phi_j. Element j corresponds to covariate j which then has a prior Dirichlet(aPhi[j],aPhi[j],....,aPhi[j]). Only used in discrete case, default=(1 1 1 ... 1).
mu0	The mean vector for mu_c in the Normal covariate case (only used in Normal covariate case, default=empirical covariate means)
Tau0	The precision matrix for mu_c in the Normal covariate case (only used in Normal covariate case, default=inverse of diagonal matrix with elements equal to squareof empirical range for each covariate)
R0	The matrix parameter for the Wishart distribution for Tau_c (only used in Normal covariate case, default=1/nCovariates * inverse of empirical covariance matrix)
kapp0	The degrees of freedom parameter for the Wishart distribution for Tau_c (only used in Normal covariate case, default=nCovariates).
muTheta	The location parameter for the t-Distribution for theta_c (only used if response included in model, default=0)
sigmaTheta	The scale parameter for the t-Distribution for theta_c (only used if response included in model, default=2.5)
dofTheta	The degrees of freedom parameter for the t-Distribution for theta_c (only used if response included in model, default=7)
muBeta	The location parameter for the t-Distribution for beta (only used when fixed effects present, default=0)

sigmaBeta	The scale parameter for the t-Distribution for beta (only used when fixed effects present, default=2.5)
dofBeta	The dof parameter for the t-Distribution for beta (only used when fixed effects present, default=7)
shapeTauEpsilon	
	Shape parameter for gamma distribution for prior for precision tau of extra variation errors epsilon (only used if extra variation is used i.e. extraYVar argument is included, default=5.0)
rateTauEpsilon	Inverse-scale (rate) parameter for gamma distribution for prior for precision tau of extra variation errors epsilon (only used if extra variation is used i.e. extraYVar argument is used, default=0.5)
aRho	Parameter for beta distribution for prior on rho in variable selection (default=0.5)
bRho	Parameter for beta distribution for prior on rho in variable selection (default=0.5)
atomRho	Parameter for the probability for the atom at zero, i.e. the 0.5 probability in w_j distributed Bernoulli(0.5) in the formulation of the sparsity inducing prior (default=0.5). This parameter must be in the interval (0,1], where atomRho=1 corresponds to the case where the prior for rho is a Beta(aRho,bRho).
shapeSigmaSqY	Shape parameter of inverse-gamma prior for sigma_Y^2 (only used in the Normal response model, default =2.5)
scaleSigmaSqY	Scale parameter of inverse-gamma prior for sigma_Y^2 (only used in the Normal response model, default =2.5)
rSlice	Slice parameter for independent slice sampler such that xi_c = (1-rSlice)*rSlice^c for c=0,1,2,... (only used for slice independent sampler i.e. sampler=SliceIndependent, default 0.75).
truncationEps	Parameter for determining the truncation level of the finite Dirichlet process (only used for truncated sampler i.e. sampler=Truncated

Value

The output of this function is a list with the components defined as above.

Authors

David Hastie, Department of Epidemiology and Biostatistics, Imperial College London, UK

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted*. Available at http://uk.arxiv.org/abs/1303.2836

Examples

```
hyp <- setHyperparams(shapeAlpha=3,rateAlpha=2,mu0=c(30,13),R0=3.2*diag(2))

inputs <- generateSampleDataFile(clusSummaryPoissonNormal())
runInfoObj<-profRegr(yModel=inputs$yModel,
    xModel=inputs$xModel, nSweeps=2, nClusInit=15,
    nBurn=2, data=inputs$inputData, output="output",
    covNames = inputs$covNames, outcomeT = inputs$outcomeT,
    fixedEffectsNames = inputs$fixedEffectNames,
    hyper=hyp)
```

---

summariseVarSelectRho   *summariseVarSelectRho*

Description

This function summarises the posterior distribution of rho, a parameter for variable selection only.

Usage

```
summariseVarSelectRho(runInfoObj)
```

Arguments

runInfoObj	Object of type runInfoObj

Value

A list with the following elements.

rho	A matrix that has as many columns as the number of covariates and as many rows as the number of sweeps. This matrix records the samples from the posterior distribution of rho for each covariate at each sweep.
rhoMean	Vector with the column means of the matrix rho above. Each value corresponds to the posterior mean of rho for each covariate.
rhoMedian	Vector with the column medians of the matrix rho above. Each value corresponds to the posterior median of rho for each covariate.
rhoLowerCI	Vector with the column lower confidence intervals of the matrix rho above. Each value corresponds to the lower confidence interval of the posterior distribution of rho for each covariate.
rhoUpperCI	Vector with the column upper confidence intervals of the matrix rho above. Each value corresponds to the upper confidence interval of the posterior distribution of rho for each covariate.

Authors

David Hastie, Department of Epidemiology and Biostatistics, Imperial College London, UK

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted*. Available at http://uk.arxiv.org/abs/1303.2836

Examples

```
inputs <- generateSampleDataFile(clusSummaryVarSelectBernoulliDiscrete())

runInfoObj<-profRegr(yModel=inputs$yModel,
    xModel=inputs$xModel, nSweeps=10, nClusInit=15,
    nBurn=20, data=inputs$inputData, output="output",
    covNames = inputs$covNames, varSelect="Continuous")

rho<-summariseVarSelectRho(runInfoObj)
```

| vec2mat | *Vector to upper triangular matrix* |

Description

Function to convert a vector to an upper triangular matrix. The vector does not include the diagonal values, which are then set equal to 1 in the matrix. The matrix is filled by row.

Usage

```
vec2mat(data = NA, nrow = 1)
```

Arguments

data	The vector to be converted, excluding the diagonal which is set equal to 1.
nrow	The number of rows (and columns) of the resulting matrix.

Value

The symmetric matrix. The matrix is filled by column.

Authors

Silvia Liverani, Department of Epidemiology and Biostatistics, Imperial College London and MRC Biostatistics Unit, Cambridge, UK

Maintainer: Silvia Liverani <liveranis@gmail.com>

References

Liverani, S., Hastie, D. I., Azizi, L., Papathomas, M. and Richardson, S. (2013) PReMiuM: An R package for Profile Regression Mixture Models using Dirichlet Processes. *Submitted*. Available at http://uk.arxiv.org/abs/1303.2836

Examples

```
vec2mat(data=c(1,2,3),nrow=3)
```

Index