Package ‘CorReg’

February 19, 2015

Type Package
Title Linear regression based on linear structure between covariates
Version 1.0
Date 2015-01-12
Maintainer Clement THERY <clement.thery@arcelormittal.com>
Description Linear regression based on a recursive structural equation model (explicit correlations) found by a MCMC algorithm. It permits to face highly correlated covariates. Variable selection is included (by lasso, elasticnet, etc.). It also provides some graphical tools for basic statistics.
License CeCILL
Copyright ArcelorMittal
URL http://www.correg.org
Depends R(>= 3.0)
Imports Rcpp(>= 0.11.0), lars(>= 1.2), Rmixmod(>= 2.0.1),
      elasticnet(>= 1.1), corplot(>= 0.73), Matrix(>= 1.1), ridge(>= 2.1), rpart(>= 4.1-5), MASS(>= 7.3-30), mvtnorm(>= 0.9),
      mclust(>= 4.2)
LinkingTo Rcpp, RcppEigen
Suggests clere(>= 1.1), spikeslab(>= 1.1.5), parcor(>= 0.2), rtkpp (>= 0.8.3)
Author Clement THERY [aut, cre],
      Christophe BIERNACKI [ctb],
      Gaetan LORIDANT [ctb],
      Florian WATRIN [ctb],
      Quentin GRIMONPREZ [ctb],
      Vincent KUBICKI [ctb],
      Samuel BLANCK [ctb],
      Jeremie KELLNER [ctb]
Repository CRAN
Repository/R-Forge/Project correg
R topics documented:

CorReg-package .................................................. 3
BicZ ................................................................. 4
BicZcurve .......................................................... 5
cleancolZ .......................................................... 5
cleanYtest .......................................................... 6
cleanZ ............................................................... 6
cleanZR2 ........................................................... 7
compare_beta ....................................................... 7
compare_sign ....................................................... 8
compare_struct .................................................... 8
compare_zero ....................................................... 9
confint_coef ....................................................... 9
correg ............................................................... 10
CVMSE .............................................................. 11
density_estimation ................................................ 11
Estep ................................................................. 12
fillmiss .............................................................. 12
hatB ................................................................. 13
matplot_zone ....................................................... 14
mixture_generator ................................................ 14
MSEZ .............................................................. 16
MSE_loc ............................................................ 16
OLS ................................................................. 17
ProbaZ ............................................................. 17
R2Z ................................................................. 18
readY ............................................................... 18
readZ ............................................................... 19
recursive_tree ...................................................... 20
rforge .............................................................. 20
searchZ_sparse .................................................... 21
showdata .......................................................... 22
structureFinder .................................................... 22
Terminator ........................................................ 23
WhoIs .............................................................. 24
Winitial ............................................................. 24
Y_generator ........................................................ 25

Index .............................................................. 26
Description

Sequential linear regression based on a structural equation model (explicit correlations). It permits to face highly correlated datasets.

Details

CorReg: see www.correg.org for article and Phd Thesis about CorReg.

Author(s)

Maintainer: Clement THERY <clement.thery@arcelormittal.com>

References

coming soon. see http://www.correg.org

Examples

# dataset generation
base=mixture_generator(n=15,p=10,ratio=0.4,tp1=1,tp2=1,tp3=1,positive=0.5,
R2Y=0.8,R2=0.9,scale=TRUE,max_compl=3,lambda=1)

X_appr=base$X_appr
Y_appr=base$Y_appr
Y_test=base$Y_test
X_test=base$X_test

TrueZ=base$Z# True generative structure (binary matrix)

# density estimation for the MCMC
density=density_estimation(X=X_appr,nbclustmax=10,detailed=TRUE)
Bic_null_vect=density$BIC_vect

# MCMC to find the structure
res=structureFinder(X=X_appr,verbose=0,reject=0,Maxiter=900,
    nbini=20,candidates=-1,Bic_null_vect=Bic_null_vect,star=TRUE,p1max=5,clean=TRUE)
hatZ=res$Z_opt
hatBic=res$bic_opt

# BIC comparison between true and found structure
bicopt_vect=BicZ(X=X_appr,Z=hatZ,Bic_null_vect=Bic_null_vect)
bicopt_vrai=BicZ(X=X_appr,Z=TRUEZ,Bic_null_vect=Bic_null_vect)
sum(bicopt_vect);sum(bicopt_vrai)

# Structure comparison
compZ=compare struct(trueZ=TRUEZ,Zalgo=hatZ)# qualitative comparison
# Interpretation of found and true structure ordered by increasing R2
readZ(Z=hatZ,crit="R2",X=X_appr,output="all",order=1)# <NA>line: name of subregressed covariate
readZ(Z=TrueZ,crit="R2",X=X_appr,output="all",order=1)# <NA> line: name of subregressed covariate

# Regression coefficients estimation
select="NULL"
resY=correg(X=X_appr,Y=Y_appr,Z=hatZ,compl=TRUE,expl=TRUE,pred=TRUE,
select=select,K=10,returning=TRUE)

MSE_complete=MSE_loc(Y=Y_test,X=X_test,intercept=TRUE,A=resy$compl$a)
MSE_expliCative=MSE_loc(Y=Y_test,X=X_test,intercept=TRUE,A=resy$expl$a)
MSE_prediCative=MSE_loc(Y=Y_test,X=X_test,intercept=TRUE,A=resy$pred$a)
MSE_vrai=MSE_loc(Y=Y_test,X=X_test,intercept=TRUE,A=base$a)

mse=data.frame(MSE_complete,MSE_expliCative,MSE_prediCative,MSE_vrai)
MSE_estimated structure
compZ$true_left;compZ$false_left

## Not run:
barplot(as.matrix(MSE),main="MSE on validation dataset", sub=paste("select=",.select))
abline(h=MSE_complete,col="red")

## End(Not run)

---

**BicZ**

**Compute the BIC of a given structure**

**Description**
Compute the BIC of a given structure

**Usage**

```r
BicZ(X = X, Z = Z, Bic_null_vect = NULL, Bic_old = NULL, methode = 1,
Zold = NULL, star = FALSE)
```

**Arguments**

- `X` : the dataset
- `Z` : binary adjacency matrix of the structure (size p)
- `Bic_null_vect` : the BIC of the null hypothesis (used for independent variables)
- `Bic_old` : BIC (vector) associated to Zold
- `methode` : parameter for OLS (matrix inversion) methode_BIC parameter for OLS (matrix inversion) 1:householderQr, 2:colPivHouseholderQr
- `Zold` : another structure with some common parts with Z (allows to compute only the differences)
- `star` : boolean defining wether classical BIC or BIC* is computed
BicZcurve

Curve of the BIC for each possible p2 with a fixed Z and truncature of Z

Description

Curve of the BIC for each possible p2 with a fixed Z and truncature of Z

Usage

BicZcurve(x = xL, Z = Z, Bic_null_vect = Bic_null_vect, plot = T, star = F, trunc = NULL)

Arguments

x
matrix containing the dataset
Z
adjacency matrix (binary) describing the structure between the variables
Bic_null_vect
vector of the BIC for each variable. used when the variable is independant
plot
boolean to plot or not the curve
star
boolean to use BIC* (hierarchical uniform law on the structure)
trunc
number of sub-regression to keep (best R2). if NULL the min of BIC is kept

cleancolZ

clean Z columns (if BIC improved)

Description

clean Z columns (if BIC improved)

Usage

cleancolZ(x = xL, Z = Z, Bic_null_vect = Bic_null_vect, methode_BIC = 1, plot = F, verbose = 1, star = FALSE)

Arguments

x
the dataset
Z
binary adjacency matrix of the structure (size p)
Bic_null_vect
vector of the BIC for each covariate
methode_BIC
parameter for OLS (matrix inversion) 1:householderQr, 2:colPivHouseholderQr
plot
if TRUE returns the vector of BIC for each step
verbose
0:none, 1:BIC,step and complexity when best BIC found 2:BIC, step, complexity, nb candidates and best candidate when best BIC found
star
if TRUE BIC* is used instead of bic
cleanYtest

Selection method based on p-values (coefficients)

Description

Selection method based on p-values (coefficients)

Usage

cleanYtest(Y = y, X = x, pvalmin = 0.05, bonferroni = F, A = NULL)

Arguments

Y  
the response variable
X  
the dataset of the covariates (without the response)
pvalmin  
the maximal bound for p-value associated to remaining coefficients
bonferroni  
boolean defining whether a Bonferroni correction is applied or not
A  
optional vector of coefficients to coerce some zeros

cleanZ

clean Z (if BIC improved)

Description

clean Z (if BIC improved)

Usage

cleanZ(X = x, Z = z, Bic_null_vect = Bic_null_vect, methode = 1,
plot = F, verbose = 1, star = FALSE)

Arguments

X  
the dataset
Z  
binary adjacency matrix of the structure (size p)
Bic_null_vect  
the BIC of the null hypothesis (used for independent variables)
methode  
parameter for OLS (matrix inversion) methode_BIC parameter for OLS (matrix inversion) 1:householderQr, 2:colPivHouseholderQr
plot  
if TRUE returns the vector of BIC for each step
verbose  
0:none, 1:BIC, step and complexity when best BIC found 2:BIC, step, complexity, nb candidates and best candidate when best BIC found
star  
boolean defining whether classical BIC or BIC* is computed
**cleanZR2**  
*To clean Z based on R2*

**Description**

To clean Z based on R2

**Usage**

```
cleanZR2(Z = Z, X = X, R2min = 0.4, methode = 1, adj = TRUE)
```

**Arguments**

- `Z`: binary adjacency matrix of the structure (size p)
- `X`: the dataset
- `R2min`: lower boundary for the structure (on R-squared value)
- `methode`: parameter for OLS (matrix inversion) 1:householderQr, 2:colPivHouseholderQr
- `adj`: boolean. Adjusted R-squared or classical one (if FALSE).

---

**compare_beta**  
*compare signs of the coefficients in two vectors*

**Description**

compare signs of the coefficients in two vectors

**Usage**

```
compare_beta(truebeta = truebeta, hatbeta = hatbeta, taux = FALSE)
```

**Arguments**

- `truebeta`: first vector
- `hatbeta`: second vector
- `taux`: boolean. Computes the ratio of each selection statistic or not.
**compare_sign**

*compare signs of the coefficients in two vectors*

**Description**

compare signs of the coefficients in two vectors

**Usage**

```r
compare_sign(trueA = trueA, Aalgo = Aalgo)
```

**Arguments**

- **trueA**: first vector
- **Aalgo**: second vector

**compare_struct**

*To compare structures (Z)*

**Description**

To compare structures (Z)

**Usage**

```r
compare_struct(trueZ = trueZ, Zalgo = Zalgo, all = TRUE, mode = "NULL")
```

**Arguments**

- **trueZ**: first structure
- **Zalgo**: second structure
- **all**: boolean. Also compute the ratio for each stat.
- **mode**: how to modify the structures before comparison. mode=c("NULL","hybrid","clique","sym")
**compare_zero**

*compare 0 values in two vectors*

**Description**

compare 0 values in two vectors

**Usage**

`compare_zero(trueA = trueA, Aalgo = Aalgo, taux = FALSE)`

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>trueA</td>
<td>first vector</td>
</tr>
<tr>
<td>Aalgo</td>
<td>second vector</td>
</tr>
<tr>
<td>taux</td>
<td>boolean. Computes the ratio of each statistic or not.</td>
</tr>
</tbody>
</table>

**confint_coef**

*plot and give confidence intervals on the coefficients estimated in a model or for proportions*

**Description**

plot and give confidence intervals on the coefficients estimated in a model or for proportions

**Usage**

`confint_coef(modele = NULL, n = NULL, prop = NULL, mean = NULL, alpha = 0.05, labels = NULL, subtitle = NULL, lang = "fr", ylim = NULL)`

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>modele</td>
<td>a model from lm on which to compute the confidence intervals</td>
</tr>
<tr>
<td>n</td>
<td>a vector of quantities associated to prop</td>
</tr>
<tr>
<td>prop</td>
<td>a vector of proportions (between 0 and 1)</td>
</tr>
<tr>
<td>mean</td>
<td>a mean value to plot</td>
</tr>
<tr>
<td>alpha</td>
<td>the risk (confidence 1-alpha)</td>
</tr>
<tr>
<td>labels</td>
<td>a vector of names to put below the bars.</td>
</tr>
<tr>
<td>subtitle</td>
<td>a subtitle to identify the graph</td>
</tr>
<tr>
<td>lang</td>
<td>if not &quot;fr&quot; then in english</td>
</tr>
<tr>
<td>ylim</td>
<td>if needed, a vector c(ymin,ymax).</td>
</tr>
</tbody>
</table>
correg

Estimates the response variable using a structure

Description
Estimates the response variable using a structure

Usage
```
correg(x = NULL, y = NULL, z = NULL, b = NULL, compl = TRUE,
expl = FALSE, pred = FALSE, select = "lar", criterion = c("MSE",
"BIC"), X_test = NULL, Y_test = NULL, intercept = TRUE, K = 10,
groupe = NULL, Amax = NULL, lambda = 1, returning = FALSE,
alpha = NULL, g = 5, compl2 = FALSE, explnew = FALSE)
```

Arguments
- **X**: the data matrix (covariates) without the intercept
- **Y**: The response variable vector
- **Z**: The structure (adjacency matrix) between the covariates
- **B**: the (p+1)xp matrix associated to Z and that contains the parameters of the sub-regressions
- **compl**: boolean to decide if the complete modele is computed
- **expl**: boolean to decide if the explicative model is in the output
- **pred**: boolean to decide if the predictive model is computed
- **select**: selection method in ("lar","lasso","forward.stagewise","stepwise", "elasticnet", "NULL","ridge","adalasso","clere","spikeslab")
- **criterion**: the criterion used to compare the models
- **X_test**: validation sample
- **Y_test**: response for the validation sample
- **intercept**: boolean. If FALSE intercept will be set to 0 in each model.
- **K**: the number of clusters for cross-validation
- **groupe**: a vector to define the groups used for cross-validation (to obtain a reproductible result)
- **Amax**: the maximum number of covariates in the final model
- **lambda**: parameter for elasticnet (quadratic penalty)
- **returning**: boolean : second predictive step (selection on I1 knowing I2 coefficients)
- **alpha**: Coefficients of the explicative model to coerce the predictive step. if not NULL explicative step is not computed.
- **g**: number of group of variables for clere
- **compl2**: boolean to compute regression (OLS only) upon [X_f,epsilon] instead of [X_f,X_r]
- **explnew**: alternative estimation
CVMSE

Cross validation

Description
Cross validation

Usage
CVMSE(x = X, y = Y, k = K, intercept = T, methode = 1,
groupe = NULL)

Arguments
- X: covariates matrix (double)
- Y: response variable
- K: number of classes
- intercept: (boolean) with or without an intercept
- methode: the method used by OLS.
- groupe: a vector to define the groups used for cross-validation (to obtain a reproducible result)

density_estimation

BIC of estimated marginal gaussian mixture densities

Description
Estimates the density of each covariate with gaussian mixture models and then gives the associated BIC.

Usage
density_estimation(x = X, nbclustmax = 10, nbclustmin = 1,
verbose = FALSE, detailed = FALSE, max = TRUE, package = c("mclust",
"Rmixmod", "rtkpp"), nbini = 20, matshape = FALSE, ...)

Arguments
- X: the dataset (matrix)
- nbclustmax: max number of clusters in the gaussian mixtures
- nbclustmin: min number of clusters in the gaussian mixtures
- verbose: verbose or not
- detailed: boolean to give the details of the mixtures found
max boolean. Use an heuristic to shrink nbclustmax according to the number of individuals in the dataset
package package to use (Rmixmod,mclust,rtkpp)
nbin number of initial points for Rmixmod
matshape boolean to give the detail in matricial shape
... additional parameters

Estep

**Imputation of missing values knowing alpha (E step of the EM)**

Description

Imputation of missing values knowing alpha (E step of the EM)

Usage

Estep(X = X, alpha = alpha, M = NULL, Z = NULL, mixmod = NULL, 
Zc = Zc, X1 = FALSE)

Arguments

X the dataset with missing values
alpha matrix of the coefficients (p+1)xp format Matrix
M binary matrix X-sized (1=missing)
Z adjacency matrix of the structure
mixmod the matrix from mixmod
Zc profil (sum) colonne de Z
X1 boolean for alternative method

fillmiss

**Fill the missing values in the dataset**

Description

Fill the missing values in the dataset

Usage

fillmiss(X = X, Z = NULL, mixmod = FALSE, B = NULL, Bt = NULL, 
res_mixmod = NULL, nbclustmax = 10, X1 = FALSE)
hatB

Arguments

X the dataset (matrix) with missing values
Z the structure associated to X. Can be a matrix of zeros if non structure.
mixmod boolean to say if the function has to use mixture model hypothesis or just the observed mean.
B the matrix of the coefficients for sub-regressions
Bt the matrix used for X1 if X1=TRUE
res_mixmod the best results found by mixmod if already computed
nbclustmax the max number of cluster for mixmod.
X1 boolean to say if dependent variables on the right will be filled based on the structure

Description

Estimates B matrix

Usage

hatB(Z = Z, X = X, methode = 1, NA_heur = FALSE)

Arguments

Z binary adjacency matrix of the structure (size p)
X the dataset
methode parameter for OLS (matrix inversion) methode_BIC parameter for OLS (matrix inversion) 1:householderQr, 2:colPivHouseholderQr
NA_heur boolean to estimate NA by the mean heuristic
matplot_zone

draws matplot with conditionnal background for easier comparison of curves.

Description
draws matplot with conditionnal background for easier comparison of curves.

Usage
matplot_zone(x = x, y = y, col = 1:6, alpha = 0.2, what = which.min,
ylim = NULL, xlim = NULL, type = "p", xlab = NULL, ylab = NULL,
main = NULL, ...)

Arguments
x the abscisses
y matrix of the curves (columns)
col list of colors (like in matplot)
alpha parameter for transparency of the background
what a function to choose a winner
ylim vector for vertical limits
xlim vector for horizontal limits
type the type of curve (like in matplot)
xlab (like in matplot)
ylab (like in matplot)
main the main title (like in matplot)
... Graphical parameters

mixture_generator

Gaussiam mixture dataset generator with regression between the covariates

Description
Gaussiam mixture dataset generator with regression between the covariates

Usage
mixture_generator(n = 130, p = 100, ratio = 0.4, max_compl = 1,
valid = 1000, positive = 0.6, sigma_Y = 10, sigma_X = NULL,
R2 = NULL, R2Y = 0.4, meanvar = NULL, sigmavar = NULL, lambda = 3,
Amax = NULL, lambdapoiss = 10, gamma = FALSE, gammashape = 1,
gammascale = 0.5, tp1 = 1, tp2 = 1, tp3 = 1, p0 = 0, nonlin = 0,
pnonlin = 2, scale = TRUE, Z = NULL)
Arguments

n the number of individuals in the learning dataset
p the number of covariates (without the response)
ratio the ratio of explained covariates (dependent)
max_compl the number of covariates in each subregression
valid the size of the validation sample
positive the ratio of positive coefficients in both the regression and the subregressions
sigma_Y standard deviation for the noise of the regression
sigma_X standard deviation for the noise of the subregression (all). ignored if gamma=T or if R2 is not NULL
R2 the strength of the subregressions
R2Y the strength of the main regression
meanvar vector of means for the covariates.
sigmavar standard deviation of the covariates.
lambda parameter of the law that define the number of components in gaussian mixture models
Amax the maximum number of covariates with non-zero coefficients in the regression
lambdapois parameter used to generate the coefficient in the subregressions. poisson distribution.
gamma boolean to generate a p-sized vector sigma_X gamma-distributed
gammashape shape parameter of the gamma distribution (if needed)
gammascale scale parameter of the gamma distribution (if needed)
tp1 the ratio of right-side covariates allowed to have a non-zero coefficient in the regression
tp2 the ratio of left-side covariates allowed to have a non-zero coefficient in the regression
tp3 the ratio of strictly independent covariates allowed to have a non-zero coefficient in the regression
pb generates Y in an heuristic way that will give some issues with correlations.
nonlin to use non linear structure (half squared , half log). if not null, it is the proba to use power pnonlin instead of log
pnonlin the power used if non linear structure
scale boolean to scale X before computing Y
Z the adjacency matrix to obtain
**MSEZ**

*Computes the MSE on the joint distribution of the dataset*

**Description**

Computes the MSE on the joint distribution of the dataset

**Usage**

\[ \text{MSEZ}(X = X, X_{\text{appr}} = \text{NULL}, B = \text{NULL}, Z = Z, \text{scale} = \text{TRUE}) \]

**Arguments**

- **X** : the dataset to predict
- **X_{\text{appr}}** : an optional learning set
- **B** : the structure tested (if known)
- **Z** : binary adjacency matrix of the structure (size \(p\))
- **scale** : boolean defining whether the dataset has to be scaled or not

**MSE_loc**

*simple MSE function*

**Description**

simple MSE function

**Usage**

\[ \text{MSE\_loc}(Y = Y, X = X, A = A, \text{intercept} = \text{T}) \]

**Arguments**

- **Y** : the response variable (vector)
- **X** : the dataset (matrix of covariates)
- **A** : the vector of coefficients
- **intercept** : boolean (to add a column of 1 to \(X\) if \(A\) contains an intercept and \(X\) don’t)
OLS

Ordinary Least Square efficiently computed with SEM for missing values

Description

Ordinary Least Square efficiently computed with SEM for missing values

Usage

OLS(X = x, Y = y, M = NULL, intercept = FALSE, sigma = FALSE, Bic = FALSE, methode = 1, miss = 0, mixmod = NULL, nbit = 100, nbclustmax = 10)

Arguments

- **x**: the covariates (double)
- **y**: the response
- **M**: binary matrix (size of X) with 1 where X is missing
- **intercept**: (boolean) is an intercept intended?
- **sigma**: (boolean) is it necessary to compute the standard deviation of errors?
- **Bic**: (boolean) is the BIC criterion computation intended?
- **methode**: parameter for OLS (matrix inversion) methode_BIC parameter for OLS (matrix inversion) 1:householderQr, 2:colPivHouseholderQr
- **miss**: to indicate wether there are missing values in X (miss=1) or not (miss=0) next version would allow missing values in Y
- **mixmod**: Gaussian Mixture hypothesis if needed. Or result of calcul_BIC_mixmod(X=X,nbclustmax=nbclustmax,bic=Bic)
- **nbit**: number of iteration for SEM
- **nbclustmax**: max number of cluster for mixmod (ignored if miss=0)

ProbaZ

Probability of Z without knowing the dataset. It also gives the exact number of binary nilpotent matrices of size p.

Description

Probability of Z without knowing the dataset. It also gives the exact number of binary nilpotent matrices of size p.

Usage

ProbaZ(Z = NULL, p = NULL, proba = FALSE, star = TRUE)
Arguments

- **Z**: binary adjacency matrix of the structure (size p)
- **p**: the number of covariates
- **proba**: gives the proba under the uniform law for Z. If `FALSE` and `star=FALSE` it gives the number of p-sized binary nilpotent matrices
- **star**: gives the log proba under uniform law for p

---

**R2Z**

*Estimates R2 of each subregression*

---

Description

Estimates R2 of each subregression

Usage

```r
R2Z(Z = zL, X = xL, methode = 1L, adj = F, crit = c("R2", "F", "sigmaX"))
```

Arguments

- **Z**: binary adjacency matrix of the structure (size p)
- **X**: the dataset
- **methode**: parameter for OLS (matrix inversion) `methode_BIC` parameter for OLS (matrix inversion) 1:householderQr, 2:colPivHouseholderQr
- **adj**: boolean to choose between adjusted R-squared and classical one
- **crit**: to choose between the R-squared and the F statistic (p-value)

---

**readY**

*a summary-like function*

---

Description

a summary-like function

Usage

```r
readY(A = A, labels = NULL, X = NULL, intercept = TRUE, ANOVA = FALSE, print = FALSE, Y = NULL)
```
**Arguments**

- **A**: coefficient vector
- **labels**: name of the covariates
- **X**: the dataset (named) if labels is null
- **intercept**: boolean defining whether A contains an intercept or not
- **ANOVA**: boolean to add Anova test for each coefficient
- **print**: boolean to print ANOVA if computed
- **Y**: the response variable if ANOVA is computed

---

**Description**

read the structure and explain it

**Usage**

```r
readZ(Z = Z, B = NULL, crit = c("none", "R2", "F", "sigmaX"),
      varnames = NULL, output = c("index", "names", "all"), X = NULL,
      order = 1)
```

**Arguments**

- **Z**: binary adjacency matrix of the structure (size p)
- **B**: is the complete structure (weighted)
- **crit**: define the criterion to use c("none", "R2", "F", "sigmaX")
- **varnames**: the names of the variables (same order)
- **output**: indicates the content of the output output=c("index","names","all")
- **X**: is a dataframe or matrix containing the dataset
- **order**: Define the order used (0: none, -1: decreasing, 1: growing)
---

title: rforge

rforge

*Upgrades a package to the lastest version on R-forge*

**Description**

Upgrades a package to the lastest version on R-forge

**Usage**

```
rforge(package = "CorReg", update = FALSE)
```

**Arguments**

- `package`: name of the packages to upgrade
- `update`: boolean to say if it is an update (detach the package)

---

recursive_tree

*decision tree in a recursive way*

**Description**

decision tree in a recursive way

**Usage**

```
recursive_tree(data = data, Y = "Y", modele = NULL, kill = NULL,
index = NULL, print = TRUE, plot = TRUE, main = NULL, sub = NULL,
lang = c("en", "fr"))
```

**Arguments**

- `data`: the dataset including the response
- `Y`: the name of the response
- `modele`: vector of the names used for the last tree search (not necessarily choosen)
- `kill`: vector of the names to kill
- `index`: to give a number to the plot
- `print`: boolean to print the tree parameters
- `plot`: boolean to plot the tree
- `main`: the main title if plot=TRUE
- `sub`: the subtitle (if NULL it is automatically added)
- `lang`: the language for the automatic subtitle in the plot
**searchZ_sparse**  

**Sparse structure research**

**Description**

Sparse structure research

**Usage**

```r
searchZ_sparse(X = x, Zi = NULL, Zj = NULL, Si = NULL, Sj = NULL, 
Bic_null_vect = NULL, candidates = 2, methode = 1, p1max = 5, 
Maxiter = 1, plot = F, best = T, better = F, random = T, 
verbose = 1, nb_opt_max = NULL)
```

**Arguments**

- **X**  
  the dataset
- **Zi**  
  indices of the rows of the 1
- **Zj**  
  indices of the columns of the 1
- **Si**  
  rowSums vector
- **Sj**  
  colSums vector
- **Bic_null_vect**  
  the BIC of the null hypothesis (used for independent variables)
- **candidates**  
  0:row and column,-1:column only, int>0:random int candidates, -2 : all (but the diag), -3 : non-zeros
- **methode**  
  parameter for OLS (matrix inversion) 1:householderQr, 2:colPivHouseholderQr
- **p1max**  
  maximum complexity for a regression
- **Maxiter**  
  number of steps
- **plot**  
  TRUE: returns for each step the type of move, complexity and BIC
- **best**  
  TRUE: systematically jumps to the best BIC seen ever when seen (it is stored even if best=FALSE)
- **better**  
  TRUE: systematically jumps to the best candidate if better than stationnarity (random weighted jump otherwise)
- **random**  
  if FALSE:moves only to improve and only to the best
- **verbose**  
  0:none, 1:BIC,step and complexity when best BIC found 2:BIC, step, complexity, nb candidates and best candidate when best BIC found
- **nb_opt_max**  
  stop criterion defining how many times the chain can walk (or stay) on the max found

**Value**

step 0:delete, 1: add, 2: stationnarity
showdata

show the missing values of a dataset

Description
show the missing values of a dataset

Usage
showdata(X = X, what = c("miss", "correl"))

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>the dataset to analyse</td>
</tr>
<tr>
<td>what</td>
<td>indicates what to plot</td>
</tr>
</tbody>
</table>

structureFinder

MCMC algo to find a structure between the covariates

Description
MCMC algo to find a structure between the covariates

Usage
structureFinder(X = X, Z = NULL, Bic_null_vect = NULL, candidates = -1,
reject = 0, methode = 1, p1max = 5, p2max = NULL, Maxiter = 1,
plot = FALSE, best = TRUE, better = FALSE, random = TRUE,
verbose = 1, nb_opt_max = NULL, exact = TRUE, nbini = NULL,
star = TRUE, clean = TRUE, ...) 

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>the dataset</td>
</tr>
<tr>
<td>Z</td>
<td>binary adjacency matrix of size p. if NULL zero matrix is used</td>
</tr>
<tr>
<td>Bic_null_vect</td>
<td>the BIC of the null hypothesis (used for independent variables)</td>
</tr>
<tr>
<td>candidates</td>
<td>0:row and column,-1:column only, int&gt;0:random int candidates,-2 : all (but the diag), -3 : non-zeros</td>
</tr>
<tr>
<td>reject</td>
<td>0: constraint relaxation, 1: reject mode</td>
</tr>
<tr>
<td>methode</td>
<td>parameter for OLS (matrix inversion) 1:householderQr, 2:colPivHouseholderQr</td>
</tr>
<tr>
<td>p1max</td>
<td>maximum complexity for a regression</td>
</tr>
<tr>
<td>p2max</td>
<td>maximum number of regressions</td>
</tr>
<tr>
<td>Maxiter</td>
<td>number of steps</td>
</tr>
</tbody>
</table>
 plot TRUE: returns for each step the type of move, complexity and BIC
best TRUE: systematically jumps to the best BIC seen ever when seen (it is stored even if best=FALSE)
better TRUE: systematically jumps to the best candidate if better than stationnarity (random weighted jump otherwise)
random if FALSE:moves only to improve and only to the best
verbose 0: none, 1: BIC, step and complexity when best BIC found 2: BIC, step, complexity, nb candidates and best candidate when best BIC found
nb_opt_max stop criterion defining how many times the chain can walk (or stay) on the max found
exact boolean. If exact subregression is found it gives its content (another verbose mode).
nbini Number of initialisations (using Winitial if Z is NULL). if NULL and Zini is NULL: only one chain beginning with zero matrix.
star boolean to compute BIC* instead of BIC (stronger penalization of the complexity). WARNING: star=TRUE implies p2max<=p/2
clean cleaning steps at the end of the walk (testing each remaining 1 for removal)
...

Value

step 0: delete, 1: add, 2: stationnarity

---

Terminator

Destructing values to have missing ones

Description

Destructing values to have missing ones

Usage

Terminator(target = NULL, wrath = 0.1, diag = 0, Z = NULL)

Arguments

target the dataset (matrix or data.frame) in which missing values will be made
wrath the ratio of missing values in the output
diag if >0 it is the thickness of the diagonal band of missing values
Z adjacency matrix to coerce a maximum of 1 missing value per sub-regression for each individual
WhoIs

*Give the partition implied by a structure*

**Description**

Give the partition implied by a structure

**Usage**

`WhoIs(Z = Z, I3 = F, I2 = T, I1 = T)`

**Arguments**

- `Z` the structure (square matrix)
- `I1, I2, I3` wanted outputs (I1= explicative covariates, I2=redundant covariates, I3=independent covariates)

Winitial

*initialization based on a weight matrix (correlation or other)*

**Description**

initialization based on a weight matrix (correlation or other)

**Usage**

`Winitial(W = W, X = NULL, p1max = NULL, BIC = F, Bic_null_vect = NULL, relax = T, random = T, nbclustmax = 5, sorted = T, mode = c("all","sorted","multinom"), p2max = NULL)`

**Arguments**

- `W` Weight matrix with values between 0 and 1 (e.g.: abs(cor(X)))
- `X` the dataset (if BIC=TRUE)
- `p1max` maximum complexity for a regression
- `BIC` boolean to use BIC for probabilities.
- `Bic_null_vect` the BIC of the null hypothesis (used for independent variables)
- `relax` TRUE: constraint relaxation, FALSE: reject mode
- `random` if FALSE W is not used.
- `nbclustmax` parameter for calcul_BIC_mixmod if needed
- `sorted` boolean to sort the candidates based on W (best first)
- `mode` candidates strategy
- `p2max` maximum number of regressions
Y_generator

Response variable generator with a linear model

Description
Response variable generator with a linear model

Usage
Y_generator(X = X, Amax = NULL, sigma_Y = 10, positive = 0.6,
Z = NULL, tp1 = 1, tp2 = 1, tp3 = 1, pb = 0)

Arguments
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>the dataset</td>
</tr>
<tr>
<td>Amax</td>
<td>maximum number of non-zero coefficients</td>
</tr>
<tr>
<td>sigma_Y</td>
<td>the standard deviation of the noise</td>
</tr>
<tr>
<td>positive</td>
<td>the ratio of positive coefficients</td>
</tr>
<tr>
<td>Z</td>
<td>the structure adjacency matrix</td>
</tr>
<tr>
<td>tp1</td>
<td>ratio of the right-side covariates in A</td>
</tr>
<tr>
<td>tp2</td>
<td>ratio of the left-side covariates in A</td>
</tr>
<tr>
<td>tp3</td>
<td>ratio of the totally independent covariates (if exists) in A</td>
</tr>
<tr>
<td>pb</td>
<td>kind of problem: 0=none, 1=simple, 2=strong</td>
</tr>
</tbody>
</table>
Index

∗Topic package

CorReg-package, 3

BicZ, 4
BicZcurve, 5

cleancolZ, 5
cleanYtest, 6
cleanZ, 6
cleanZR2, 7
compare_beta, 7
compare_sign, 8
compare_struct, 8
compare_zero, 9
confint_coef, 9
correg, 10
CorReg-package, 3
CVMSE, 11
density_estimation, 11
Estep, 12
fillmiss, 12
hatB, 13

corregMpackage, 3

cvmse, 11
density_estimation, 11
Estep, 12
fillmiss, 12
hatB, 13

matplot_zone, 14
mixture_generator, 14
MSE_loc, 16
MSEZ, 16

OLS, 17
ProbaZ, 17
R2Z, 18
readY, 18
readZ, 19
recursive_tree, 20
rforge, 20

searchZ_sparse, 21
showdata, 22
structureFinder, 22
Terminator, 23
WhoIs, 24
Winitial, 24
Y_generator, 25