Package 'clere'

February 6, 2020

Type Package

Title Simultaneous Variables Clustering and Regression

Version 1.2.0

Description Implements an empirical Bayes approach for simultaneous variable clustering and regression. This version also (re)implements in C++ an R script proposed by Howard Bondell that fits the Pairwise Absolute Clustering and Sparsity (PACS) methodology (see Sharma et al (2013) <DOI:10.1080/15533174.2012.707849>).

License GPL-3

URL https://github.com/mcanouil/clere

BugReports https://github.com/mcanouil/clere/issues

Depends R (>= 3.5.0)

LinkingTo Rcpp, RcppEigen

Imports Rcpp (>= 1.0.0), graphics, methods, parallel, utils

Suggests covr (>= 3.4.0), knitr (>= 1.26), lasso2 (>= 1.2.20)

VignetteBuilder knitr

Copyright Loic Yengo

Encoding UTF-8

LazyData true

NeedsCompilation yes

Collate 'algoComp.R' 'numExpRealData.R' 'numExpSimData.R' 'fitClere.R' 'Clere-class.R' 'sClere-class.R' 'fitPacs.R' 'Pacs-class.R' 'clere-package.R'

Author Loic Yengo [aut, cre],
Mickaël Canouil [aut] (<https://orcid.org/0000-0002-3396-4549>)

Maintainer Loic Yengo <loic.yengo@gmail.com>

Repository CRAN

Date/Publication 2020-02-06 19:30:02 UTC

R topics documented:

`clere-package` *CLERE methodology for simultaneous variables clustering and regression*

Description

The methodology consists in creating clusters of variables involved in a high dimensional linear regression model so as to reduce the dimensionality. A model-based approach is proposed and fitted using a Stochastic EM-Gibbs algorithm (SEM-Gibbs).

See Also

Overview : `clere-package`
Classes : `Clere`, `Pacs`
Methods : `plot`, `clusters`, `predict`, `summary`
Functions : `fitClere`, `fitPacs` Datasets : `numExpRealData`, `numExpSimData`, `algoComp`

Examples

```
# Simple example using simulated data
# to see how to you the main function clere
library(clere)
x  <- matrix(rnorm(50 * 100), nrow = 50, ncol = 100)
y  <- rnorm(50)
model <- fitClere(y = y, x = x, g = 2, plotit = FALSE)
plot(model)
clus <- clusters(model, threshold = NULL)
predict(model, newx = x+1)
summary(model)
```

algoComp *Performances SEM algorithm versus MCEM*

Description

This data contains four matrices corresponding to four performance indictors used to compare SEM algorithm and three versions of the MCEM algorithm (MCEMA: with 5 MC interations; MCEMB: with 25 MC iterations and MCEMC: 125 MC iterations) as described in the package vignette. The first matrix `Pred` contains prediction errors; matrix `Bias` contains the bias over all model parameters, matrix `Time` contains execution times for the four methods and matrix `Liks` the log-likelihood reached by each method. These data were used to generate the Table 1. in the package vignette. For more details, please refer to the package vignette. The R script used to create this dataset is clere/inst/doc/SEM_vs_MCEM_simulations.R.

Usage

```
algoComp
```

Format

A list containing four 200 x 4/5 matrices.

See Also

Overview : `clere-package`
Classes : `Clere`, `Pacs`
Methods : `plot`, `clusters`, `predict`, `summary`
Functions : `fitClere`, `fitPacs` Datasets : `numExpRealData`, `numExpSimData`, `algoComp`

clusters *clusters method*

Description

This function makes returns the estimated clustering of variables.

Usage

```
clusters(object, threshold = NULL, ...)
```

Arguments

`object`	[Clere]: Output object from `fitClere`.
`threshold`	[numeric]: A numerical `threshold` > `0.5`. If `threshold` = `NULL` then the each variable is assigned to the cluster having the largest associated posterior probability.
`...`	Additional arguments, not to be supplied in this version.

See Also

Overview : clere-package
Classes : Clere
Methods : show, plot, clusters, predict, summary
Functions : fitClere Datasets : numExpRealData, numExpSimData

fitClere *fitClere function*

Description

This function runs the CLERE Model. It returns an object of class Clere. For more details please refer to clere.

Usage

```
fitClere(
  y,
  x,
  g = 1,
  nItMC = 50,
  nItEM = 1000,
  nBurn = 200,
  dp = 5,
  nsamp = 200,
  maxit = 500,
  tol = 0.001,
  nstart = 2,
  parallel = FALSE,
  seed = NULL,
  plotit = FALSE,
  sparse = FALSE,
  analysis = "fit",
  algorithm = "SEM",
  theta0 = NULL,
  Z0 = NULL
)
```

Arguments

`y` [numeric]: The vector of observed responses - size `n`.

`x` [matrix]: The matrix of predictors - size `n` rows and `p` columns.

`g` [integer]: Either the number or the maximum of groups for fitting CLERE. Maximum number of groups is considered when model selection is required.

`nItMC` [numeric]: Number of Gibbs iterations to generate the partitions. After the `nBurn` iterations, this number is automatically set to `1`.

`nItEM` [numeric]: Number of SEM iterations.

`nBurn` [numeric]: Number of SEM iterations discarded before calculating the MLE which is averaged over SEM draws.

`dp` [numeric]: Number of iterations between sampled partitions when calculating the likelihood at the end of the run.

`nsamp` [numeric]: Number of sampled partitions for calculating the likelihood at the end of the run.

`maxit` [numeric]: An EM algorithm is used inside the SEM to maximize the complete log-likelihood p(y, Z|theta). `maxit` stands as the maximum number of EM iterations for the internal EM.

`tol` [numeric]: Maximum increased in complete log-likelihood for the internal EM (stopping criterion).

`nstart` [integer]: Number of random starting points to be used for fitting the model.

`parallel` [logical]: Should the estimation from `nstart` random starting points run in parallel?

`seed` [integer]: An integer given as a seed for random number generation. If set to `NULL`, then a random seed is generated between `1` and `1000`.

`plotit` [logical]: Should a summary plot (base plot) be drawn after the run?

`sparse` [logical]: Should a `0` class be imposed to the model?

`analysis` [character]: Which analysis is to be performed. Values are `"fit"`, `"bic"`, `"aic"` and `"icl"`.

`algorithm` [character]: The algorithm to be chosen to fit the model. Either the SEM-Gibbs algorithm or the MCEM algorithm. The most efficient algorithm being the SEM-Gibbs approach. MCEM is not available for binary response.

`theta0` [vector(numeric)]: An initial guess of the model parameters. When considering g components, the length of `theta0` must be `2*g+3` and `theta0` should be filled as intercept, the b_k's (g real numbers), the pi_k's (g real numbers summing to 1), sigma^2 and gamma^2 (two positive numbers).

`Z0` [vector(integer)]: A vector of integers representing an initial partition for the variables. For 10 variables and 3 groups `Z0` can be defined as \ `Z0 = c(rep(0,2),rep(1,3),rep(2,5))`.

Value

Object of class `Clere`.

See Also

Overview : `clere-package`
Classes : `Clere`, `Pacs`
Methods : `plot`, `clusters`, `predict`, `summary`
Functions : `fitClere`, `fitPacs` Datasets : `numExpRealData`, `numExpSimData`, `algoComp`

Examples

```
library(clere)
plotit    <- FALSE
sparse    <- FALSE
nItEM     <- 100
nBurn     <- nItEM / 2
nsamp     <- 100
analysis  <- "fit"
algorithm <- "SEM"
nItMC     <- 1
dp        <- 2
maxit     <- 200
tol       <- 1e-3

n         <- 50
p         <- 50
intercept <- 0
sigma     <- 10
gamma     <- 10
rho       <- 0.5

g         <- 5
probs     <- c(0.36, 0.28, 0.20, 0.12, 0.04)
Eff       <- p * probs
a         <- 5
B         <- a**(0:(g-1))-1
Z         <- matrix(0, nrow = p, ncol = g)
imax      <- 0
imin      <- 1

for (k in 1:g) {
    imin <- imax+1
    imax <- imax+Eff[k]
    Z[imin:imax, k] <- 1
}
Z <- Z[sample(1:p, p), ]
if (g>1) {
    Beta <- rnorm(p, mean = c(Z%*%B), sd = gamma)
} else {
    Beta <- rnorm(p, mean = B, sd = gamma)
}

theta0 <- NULL # c(intercept, B, probs, sigma^2, gamma^2)
Z0     <- NULL # apply(Z, 1, which.max)-1

gmax <- 7

## Prediction
eps  <- rnorm(n, mean = 0, sd = sigma)
X    <- matrix(rnorm(n*p), nrow = n, ncol = p)
Y    <- as.numeric(intercept+X%*%Beta+eps)
```

```
tt   <- system.time(mod <- fitClere(y = Y, x = X, g = gmax,
                        analysis = analysis,algorithm = algorithm,
                        plotit = plotit,
                        sparse = FALSE,nItEM = nItEM,
                        nBurn = nBurn, nItMC = nItMC,
                        nsamp = nsamp, theta0 = theta0, Z0 = Z0) )
plot(mod)
Yv <- predict(object = mod, newx = X)
```

fitPacs *fitPacs function*

Description

This function implements the PACS (Pairwise Absolute Clustering and Sparsity) methodology of Sharma DB et al. (2013). This methodology proposes to estimate the regression coefficients by solving a penalized least squares problem. It imposes a constraint on Beta (the vector of regression coefficients) that is a weighted combination of the L1 norm and the pairwise L-infinity norm. Upper-bounding the pairwise L-infinity norm enforces the covariates to have close coefficients. When the constraint is strong enough, closeness translates into equality achieving thus a grouping property. For PACS, no software was available. Only an R script was released on Bondell's webpage (http://www4.stat.ncsu.edu/~bondell/Software/PACS/PACS.R.r). Since this R script was running very slowly, we decided to reimplement it in C++ and interfaced it with the present R package clere. This corresponds to the option `type=1` in Bondell's script.

Usage

```
fitPacs(Y, X, lambda = 0.5, betaInput, epsPACS = 1e-05, nItMax = 1000)
```

Arguments

`Y` [numeric]: The vector of observed responses - size `n`.

`X` [matrix]: The matrix of predictors - size `n` rows and `p` columns.

`lambda` [numeric]: A non-negative penalty term that controls simultaneouly clusetering and sparsity.

`betaInput` [numeric]: A vector of initial guess of the model parameters. The authors suggest to use coefficients obtained after fitting a ridge regression with the shrinkage parameter selected using AIC criterion.

`epsPACS` [numeric]: A tolerance threshold that control the convergence of the algorithm. The default value fixed in Bondell's initial script is 1e-5.

`nItMax` [numeric]: Maximum number of iterations in the algorithm.

Value

Object of class `Pacs` containing all the input parameters plus parameter `a0` the intercept and parameter `K` the dimensionality of the model.

See Also

Overview : clere-package
Classes : Clere, Pacs
Methods : plot, clusters, predict, summary
Functions : fitClere, fitPacs Datasets : numExpRealData, numExpSimData, algoComp

Examples

```
n     <- 100
p     <-  20
Beta  <- rep(c(0,2),10)
eps   <- rnorm(n,sd=3)
x     <- matrix(rnorm(n*p), nrow = n, ncol = p)
y     <- as.numeric(10+x%*%Beta+eps)
bInit <- lm(y~scale(x))$coefficients[-1]
mod   <- fitPacs(Y=y,X=x,lambda=1.25,betaInput=bInit,epsPACS=1e-5,nItMax=1000)
```

---

numExpRealData *Performances of 9 methods for dimension reduction applied to 2 published real dataset*

Description

This data contains two matrices: one for the Prostate dataset (from R package lasso2) and the other for the eyedata dataset (from R package flare). Each matrix has 5 rows and 28 colums. The columns can be grouped as three blocs of 9 (for each method compared: LASSO, RIDGE, Elastic net [ELNET], Stepwise variable selection [STEP], CLERE, CLERE sparse [CLERE_s], Spike and Slab [SS], AVG method and Pairwise Absolute Clustering and Sparsity [PACS]). The 1st 9 (1:9) contain prediction error obtained by 5-fold cross validation using 10 random permutation of the covariate matrix. The 2nd 9 columns (10:18) contain the number of parameters estimated for each method. The 3rd 9 columns are times in seconds measured for fitting each methods. The 28 column is the seed utilized for generating random numbers in these analyses. For more details, please refer the package vignette. The R script used to create this dataset is clere/inst/doc/RealDataExample.R.

Usage

```
numExpRealData
```

Format

A list containing two matrices: one for the Prostate dataset (from R package lasso2) and the other for the Eye dataset (from R package flare)

See Also

Overview : clere-package
Classes : Clere, Pacs
Methods : plot, clusters, predict, summary
Functions : fitClere, fitPacs Datasets : numExpRealData, numExpSimData, algoComp

---

numExpSimData *Performances of 9 methods for dimension reduction on data simulated under the CLERE model*

Description

This dataset is a matrix of 200 rows and 28 colums. The columns can be grouped as three blocs of 9 (for each method compared: LASSO, RIDGE, Elastic net [ELNET], Stepwise variable selection [STEP], CLERE, CLERE sparse [CLERE_s], Spike and Slab [SS], AVG method and Pairwise Absolute Clustering and Sparsity [PACS]). Prediction errors (MSE), number of estimated parameters and time (seconds) to fit the data are compared.The 1st 9 (1:9) contain prediction error obtained by 5-fold cross validation using 10 random permutation of the covariate matrix. The 2nd 9 columns (10:18) contain the number of parameters estimated for each method. The 3rd 9 columns are times in seconds measured for fitting each methods. The 28 column is the seed utilized for generating random numbers in these analyses. Each row corresponds to a simulated dataset on which all 9 methods were fitted. For more details, please refer to the package vignette. The R script used to create this dataset is clere/inst/doc/SimulatedDataExample.R.

Usage

```
numExpSimData
```

Format

A 200 x 28 matrix.

See Also

Overview : clere-package
Classes : Clere, Pacs
Methods : plot, clusters, predict, summary
Functions : fitClere, fitPacs Datasets : numExpRealData, numExpSimData, algoComp

Pacs-class *Pacs class*

Description

This class contains all the input parameters to run CLERE.

Details

Y [numeric]: The vector of observed responses - size `n`.

X [matrix]: The matrix of predictors - size `n` rows and `p` columns.

lambda [numeric]: A non-negative penalty term that controls simultaneouly clusetering and sparsity.

betaInput [numeric]: A vector of initial guess of the model parameters. The authors suggest to use coefficients obtained after fitting a ridge regression with the shrinkage parameter selected using AIC criterion.

epsPACS [numeric]: A tolerance threshold that control the convergence of the algroithm. The default value fixed in Bondell's initial script is 1e-5.

nItMax [integer]: Maximum number of iterations in the algorithm.

a0 [numeric]: Fitted intercept.

K [integer]: Model dimensionality.

Methods

object["slotName" :] Get the value of the field `slotName`.

object["slotName" <-value:] Set `value` to the field `slotName`.

See Also

Overview : `clere-package`
Classes : `Clere`, `Pacs`
Methods : `plot`, `clusters`, `predict`, `summary`
Functions : `fitClere`, `fitPacs` Datasets : `numExpRealData`, `numExpSimData`, `algoComp`

plot-methods *plot method*

Description

Graphical summary for MCEM/SEM-Gibbs estimation. This function represents the course of the model parameters in view of the iterations of the estimation algorithms implemented in `fitClere`.

Usage

```
## S4 method for signature 'Clere'
plot(x, y, ...)
```

Arguments

`x` [Clere]: Output object from `fitClere`.

`y` [any]: Unused parameter.

`...` Additional arguments, not to be supplied in this version.

See Also

Overview : `clere-package`
Classes : `Clere`, `Pacs`
Methods : `plot`, `clusters`, `predict`, `summary`
Functions : `fitClere`, `fitPacs` Datasets : `numExpRealData`, `numExpSimData`, `algoComp`

`predict` *predict method*

Description

This function makes prediction using a fitted model and a new matrix of design. It returns a vector of predicted values of size equal to the number of rows of matrix newx.

Usage

```
## S4 method for signature 'Clere'
predict(object, newx, ...)
```

Arguments

`object` [Clere]: Output object from `fitClere`.

`newx` [matrix]: A numeric design matrix.

`...` Additional arguments, not to be supplied in this version.

See Also

Overview : `clere-package`
Classes : `Clere`
Methods : `show`, `plot`, `clusters`, `predict`, `summary`
Functions : `fitClere` Datasets : `numExpRealData`, `numExpSimData`

summary *summary method*

Description

This function summarizes the output of function fitClere.

Usage

```
## S4 method for signature 'Clere'
summary(object, ...)
```

Arguments

object	[Clere]: Output object from fitClere.
...	Additional arguments, not to be supplied in this version.

See Also

Overview : clere-package
Classes : Clere
Methods : show, plot, clusters, predict, summary
Functions : fitClere Datasets : numExpRealData, numExpSimData

Index