Package 'msir'

October 13, 2022

Version 1.3.3

Date 2020-12-15

Title Model-Based Sliced Inverse Regression

Description An R package for dimension reduction based on finite Gaussian mixture modeling of inverse regression.

Depends R (>= 3.0)

Imports mclust (>= 5.4), stats, utils, graphics, grDevices

Suggests knitr (>= 1.20), rmarkdown (>= 1.10), rgl (>= 0.100)

License GPL (>= 2)

VignetteBuilder knitr

URL https://mclust-org.github.io/msir/

Repository CRAN

ByteCompile true

LazyData yes

Encoding UTF-8

NeedsCompilation no

Author Luca Scrucca [aut, cre] (<https://orcid.org/0000-0003-3826-0484>)

Maintainer Luca Scrucca <luca.scrucca@unipg.it>

Date/Publication 2020-12-16 12:20:02 UTC

R topics documented:

`msir-package` *Model-based Sliced Inverse Regression (MSIR)*

Description

An R package that implements MSIR, a dimension reduction method based on Gaussian finite mixture models. The basis of the subspace is estimated by modeling the inverse distribution within slice using finite mixtures of Gaussians, with number of components and covariance matrix parameterization selected by BIC or defined by the user. The method provides an extension to sliced inverse regression (SIR) and allows to overcome the main limitation of SIR, i.e., the failure in the presence of regression symmetric relationships, without the need to impose further assumptions.

Author(s)

Luca Scrucca <luca.scrucca@unipg.it>

References

Scrucca, L. (2011) Model-based SIR for dimension reduction. *Computational Statistics & Data Analysis*, 55(11), 3010-3026.

See Also

`msir`

`loess.sd` *Local Polynomial Regression Fitting with Variability bands*

Description

Nonparametric estimation of mean function with variability bands.

Usage

```
loess.sd(x, y = NULL, nsigma = 1, ...)

panel.loess(x, y, col = par("col"), bg = NA, pch = par("pch"), cex = 1,
            col.smooth = "red", span = 2/3, degree = 2, nsigma = 1, ...)
```

Arguments

x	a vector of values for the predictor variable x.
y	a vector of values for the response variable y.
nsigma	a multiplier for the standard deviation function.
col, bg, pch, cex	numeric or character codes for the color(s), point type and size of points; see also par.
col.smooth	color to be used by lines for drawing the smooths.
span	smoothing parameter for loess.
degree	the degree of the polynomials to be used, see loess.
...	further argument passed to the function loess.

Value

The function loess.sd computes the loess smooth for the mean function and the mean plus and minus k times the standard deviation function.

The function panel.loess can be used to add to a scatterplot matrix panel a smoothing of mean function using loess with variability bands at plus and minus nsigmas times the standard deviation.

Author(s)

Luca Scrucca <luca.scrucca@unipg.it>

References

Weisberg, S. (2005) Applied Linear Regression, 3rd ed., Wiley, New York, pp. 275-278.

See Also

loess

Examples

```
data(cars)
plot(cars, main = "lowess.sd(cars)")
lines(l <- loess.sd(cars))
lines(l$x, l$upper, lty=2)
lines(l$x, l$lower, lty=2)
```

`msir` *Model-based Sliced Inverse Regression (MSIR)*

Description

A dimension reduction method based on Gaussian finite mixture models which provides an extension to sliced inverse regression (SIR). The basis of the subspace is estimated by modeling the inverse distribution within slice using Gaussian finite mixtures with number of components and covariance matrix parameterization selected by BIC or defined by the user.

Usage

```
msir(x, y, nslices = msir.nslices, slice.function = msir.slices,
     modelNames = NULL, G = NULL, cov = c("mle", "regularized"), ...)
```

Arguments

`x`	A $(n \times p)$ design matrix containing the predictors data values.
`y`	A $(n \times 1)$ vector of data values for the response variable. It can be a numeric vector (regression) but also a factor (classification). In the latter case, the levels of the factor define the slices used.
`nslices`	The number of slices used, unless `y` is a factor. By default the value returned by `msir.nslices`.
`slice.function`	The slice functions to be used, by default `msir.slices`, but the user can provide a different slicing function.
`modelNames`	A vector of character strings indicating the Gaussian mixture models to be fitted as described in `mclustModelNames`. If a vector of strings is given they are used for all the slices. If a list of vectors is provided then each vector refers to a single slice.
`G`	An integer vector specifying the numbers of mixture components used in fitting Gaussian mixture models. If a list of vectors is provided then each vector refers to a single slice.
`cov`	The predictors marginal covariance matrix. Possible choices are: • `"mle"`: for the maximum likelihood estimate • `"regularized"`: for a regularized estimate of the covariance matrix (see `msir.regularizedSigma`) • R `matrix`: a $(p \times p)$ user defined covariance matrix
`...`	other arguments passed to `msir.compute`.

Value

Returns an object of class `'msir'` with attributes:

`call`	the function call.

x	the design matrix.
y	the response vector.
slice.info	output from slicing function.
mixmod	a list of finite mixture model objects as described in mclustModel.
loglik	the log-likelihood for the mixture models.
f	a vector of length equal to the total number of mixture components containing the fraction of observations in each fitted component within slices.
mu	a matrix of component within slices predictors means.
sigma	the marginal predictors covariance matrix.
M	the msir kernel matrix.
evalues	the eigenvalues from the generalized eigen-decomposition of M.
evectors	the raw eigenvectors from the generalized eigen-decomposition of M ordered according to the eigenvalues.
basis	the normalized eigenvectors from the generalized eigen-decomposition of M ordered according to the eigenvalues.
std.basis	standardized basis vectors obtained by multiplying each coefficient of the eigenvectors by the standard deviation of the corresponding predictor. The resulting coefficients are scaled such that all predictors have unit standard deviation.
numdir	the maximal number of directions estimated.
dir	the estimated MSIR directions from mean-centered predictors.

Author(s)

Luca Scrucca <luca.scrucca@unipg.it>

References

Scrucca, L. (2011) Model-based SIR for dimension reduction. *Computational Statistics & Data Analysis*, 55(11), 3010-3026.

See Also

summary.msir, plot.msir.

Examples

```
# 1-dimensional simple regression
n <- 200
p <- 5
b <- as.matrix(c(1,-1,rep(0,p-2)))
x <- matrix(rnorm(n*p), nrow = n, ncol = p)
y <- exp(0.5 * x%*%b) + 0.1*rnorm(n)
MSIR <- msir(x, y)
summary(MSIR)
plot(MSIR, type = "2Dplot")
```

```
# 1-dimensional symmetric response curve
n <- 200
p <- 5
b <- as.matrix(c(1,-1,rep(0,p-2)))
x <- matrix(rnorm(n*p), nrow = n, ncol = p)
y <- (0.5 * x%*%b)^2 + 0.1*rnorm(n)
MSIR <- msir(x, y)
summary(MSIR)
plot(MSIR, type = "2Dplot")
plot(MSIR, type = "coefficients")

# 2-dimensional response curve
n <- 300
p <- 5
b1 <- c(1, 1, 1, rep(0, p-3))
b2 <- c(1,-1,-1, rep(0, p-3))
b <- cbind(b1,b2)
x <- matrix(rnorm(n*p), nrow = n, ncol = p)
y <- x %*% b1 + (x %*% b1)^3 + 4*(x %*% b2)^2 + rnorm(n)
MSIR <- msir(x, y)
summary(MSIR)
plot(MSIR, which = 1:2)
## Not run: plot(MSIR, type = "spinplot")
plot(MSIR, which = 1, type = "2Dplot", span = 0.7)
plot(MSIR, which = 2, type = "2Dplot", span = 0.7)
```

`msir.bic` *BIC-type criterion for dimensionality*

Description

BIC-type criterion for selecting the dimensionality of a dimension reduction subspace.

Usage

```
msir.bic(object, type = 1, plot = FALSE)

bicDimRed(M, x, nslices, type = 1, tol = sqrt(.Machine$double.eps))
```

Arguments

`object`	a `'msir'` object
`plot`	if `TRUE` a plot of the criterion is shown.
`M`	the kernel matrix. See details below.
`x`	the predictors data matrix. See details below.
`type`	See details below.
`nslices`	the number of slices. See details below.
`tol`	a tolerance value

Details

This BIC-type criterion for the determination of the structural dimension selects d as the maximizer of

$$G(d) = l(d) - Penalty(p, d, n)$$

where $l(d)$ is the log-likelihood for dimensions up to d, p is the number of predictors, and n is the sample size. The term $Penalty(p, d, n)$ is the type of penalty to be used:

- `type = 1`: $Penalty(p, d, n) = -(p - d)\log(n)$
- `type = 2`: $Penalty(p, d, n) = 0.5Cd(2p-d+1)$, where $C = (0.5\log(n)+0.1n^(1/3))/2nslices/n$
- `type = 3`: $Penalty(p, d, n) = 0.5Cd(2p - d + 1)$, where $C = \log(n)nslices/n$
- `type = 4` $Penalty(p, d, n) = 1/2d\log(n)$

Value

Returns a list with components:

`evalues`	eigenvalues
`l`	log-likelihood
`crit`	BIC-type criterion
`d`	selected dimensionality

The `msir.bic` also assign the above information to the corresponding `'msir'` object.

Author(s)

Luca Scrucca <luca.scrucca@unipg.it>

References

Zhu, Miao and Peng (2006) "Sliced Inverse Regression for CDR Space Estimation", JASA.
Zhu, Zhu (2007) "On kernel method for SAVE", Journal of Multivariate Analysis.

See Also

`msir`

Examples

```
# 1-dimensional symmetric response curve
n <- 200
p <- 5
b <- as.matrix(c(1,-1,rep(0,p-2)))
x <- matrix(rnorm(n*p), nrow = n, ncol = p)
y <- (0.5 * x%*%b)^2 + 0.1*rnorm(n)
MSIR <- msir(x, y)
msir.bic(MSIR, plot = TRUE)
summary(MSIR)
msir.bic(MSIR, type = 3, plot = TRUE)
summary(MSIR)
```

msir.nslices *Default number of slices*

Description

This function computes a Sturges' type number of slices to be used as default in the msir function.

Usage

```
msir.nslices(n, p)
```

Arguments

n	the number of observations in the sample.
p	the number of predictors in the sample.

Value

The function returns a single value, i.e. the number of slices.

Author(s)

Luca Scrucca <luca.scrucca@unipg.it>

See Also

msir

msir.permutation.test *Permutation test for dimensionality*

Description

Approximates marginal dimension test significance levels by sampling from the permutation distribution.

Usage

```
msir.permutation.test(object, npermute = 99, numdir = object$numdir, verbose = TRUE)
```

Arguments

object	a 'msir' object.
npermute	number of permutations to compute.
numdir	maximum value of the dimension to test.
verbose	if TRUE a textual progress bar is shown during computation.

Details

The function approximates significance levels of the marginal dimension tests based on a permutation test.

Value

The function returns a list with components:

`summary`	a table containing the hypotheses, the test statistics, the permutation p-values.
`npermute`	the number of permutations used.

Furthermore, it also assigns the above information to the corresponding `'msir'` object.

Author(s)

Luca Scrucca <luca.scrucca@unipg.it>

References

Scrucca, L. (2011) Model-based SIR for dimension reduction. *Computational Statistics & Data Analysis*, 55(11), 3010-3026.

See Also

Function `dr()` in package **dr**.

Examples

```
## Not run:
# 1-dimensional simple regression
n <- 200
p <- 5
b <- as.matrix(c(1,-1,rep(0,p-2)))
x <- matrix(rnorm(n*p), nrow = n, ncol = p)
y <- exp(0.5 * x%*%b) + 0.1*rnorm(n)
MSIR <- msir(x, y)
msir.permutation.test(MSIR)
summary(MSIR)

## End(Not run)
```

`msir.regularizedSigma` *Regularized estimate of predictors covariance matrix.*

Description

This function computes a regularized version of the covariance matrix of the predictors. Among the possible models the one which maximizes BIC is returned.

Usage

```
msir.regularizedSigma(x, inv = FALSE, model = c("XII", "XXI", "XXX"))
```

Arguments

`x`	Ahe predictors data matrix.
`inv`	A logical specifying what must be returned. If `TRUE` the inverse of the estimated covariance matrix is returned, otherwise the estimated covariance matrix (default).
`model`	A character string specifying the available models: • `XII`: diagonal equal variances • `XXI`: diagonal unequal variances • `XXX`: full covariance matrix

Value

A $(p \times p)$ covariance matrix estimate.

Author(s)

Luca Scrucca <luca.scrucca@unipg.it>

See Also

`msir`

msir.slices *Slice a vector into slices of approximately equal size*

Description

Function used for slicing a continuous response variable.

Usage

```
msir.slices(y, nslices)
```

Arguments

y	a vector of n values
nslices	the number of slices, no larger than n

Value

Returns a list with components:

slice.indicator	an indicator variable for the slices.
nslices	the actual number of slices produced.
slice.sizes	the number of observations in each slice.

Author(s)

Luca Scrucca <luca.scrucca@unipg.it>

See Also

msir

plot.msir *Plot method for 'msir' objects.*

Description

Plots directions and other information from MSIR estimation.

Usage

```
## S3 method for class 'msir'
plot(x, which,
     type = c("pairs", "2Dplot", "spinplot", "evalues", "coefficients"),
     span = NULL, std = TRUE, ylab, xlab, restore.par = TRUE, ...)
```

Arguments

x	a 'msir' object.
which	a vector of value(s) giving the directions for which the plot should be drawn.
type	the type of plot to be drawn.
span	the span of smoother (only for type = "pairs" | "2Dplot").
std	if TRUE coefficients are standardized (only for type = "coefficients").
ylab	a character string for the y-axis label.
xlab	a character string for the x-axis label.
restore.par	if TRUE the graphical parameters (see par) changed are restored to the previous state. If you want to manipulate the resulting plot you should set restore.par = FALSE.
...	additional arguments.

Author(s)

Luca Scrucca <luca.scrucca@unipg.it>

References

Scrucca, L. (2011) Model-based SIR for dimension reduction. *Computational Statistics & Data Analysis*, 55(11), 3010-3026.

See Also

msir

Examples

```
## Not run:
# 2-dimensional response curve
n <- 300
p <- 5
b1 <- c(1, 1, 1, rep(0, p-3))
b2 <- c(1,-1,-1, rep(0, p-3))
b <- cbind(b1,b2)
x <- matrix(rnorm(n*p), nrow = n, ncol = p)
y <- x %*% b1 + (x %*% b1)^3 + 4*(x %*% b2)^2 + rnorm(n)
MSIR <- msir(x, y)
summary(MSIR)
plot(MSIR)
plot(MSIR, which = 1:2)
plot(MSIR, type = "2Dplot", which = 1, span = 0.7)
plot(MSIR, type = "2Dplot", which = 2, span = 0.7)
plot(MSIR, type = "spinplot")
plot(MSIR, type = "evalues")
plot(MSIR, type = "coefficients")

## End(Not run)
```

predict.msir *Model-based Sliced Inverse Regression directions*

Description

MSIR estimates a set of $d \leq p$ orthogonal direction vectors of length p which are estimates of the basis of the dimensional reduction subspace.

Usage

```
  ## S3 method for class 'msir'
predict(object, dim = 1:object$numdir, newdata, ...)
```

Arguments

object	an object of class 'msir' resulting from a call to msir.
dim	the dimensions of the reduced subspace used for prediction.
newdata	a data frame or matrix giving the data. If missing the data obtained from the call to msir are used.
...	further arguments passed to or from other methods.

Value

The function returns a matrix of points projected on the subspace spanned by the estimated basis vectors.

Author(s)

Luca Scrucca <luca.scrucca@unipg.it>

References

Scrucca, L. (2011) Model-based SIR for dimension reduction. *Computational Statistics & Data Analysis*, 55(11), 3010-3026.

See Also

{msir}

Examples

```
n <- 200
p <- 5
b <- as.matrix(c(1,-1,rep(0,p-2)))
x <- matrix(rnorm(n*p), nrow = n, ncol = p)
y <- exp(0.5 * x%*%b) + 0.1*rnorm(n)
pairs(cbind(y,x), gap = 0)
```

```
MSIR <- msir(x, y)
summary(MSIR)
plot(MSIR, which = 1, type = "2Dplot")
all.equal(predict(MSIR), MSIR$dir)
predict(MSIR, dim = 1:2)

x0 <- matrix(rnorm(n*p), nrow = n, ncol = p)
y0 <- exp(0.5 * x0%*%b) + 0.1*rnorm(n)
plot(predict(MSIR, dim = 1, newdata = x0), y0)
```

`spinplot` *Rotating three-dimensional plot*

Description

General function to draw a rgl-based rotating 3D scatterplot.

Usage

```
spinplot(x, y, z,
         scaling = c("abc", "aaa"),
         rem.lin.trend = FALSE,
         uncor.vars = FALSE,
         fit.ols = FALSE,
         fit.smooth = FALSE,
         span = 0.75,
         ngrid = 25,
         markby,
         pch.points = 1,
         col.points = "black",
         cex.points = 1,
         col.axis   = "gray50",
         col.smooth = "limegreen",
         col.ols    = "lightsteelblue",
         background = "white",
         ...)
```

Arguments

`x`	a vector of values for the variable in the horizontal (H) screen axis.
`y`	a vector of values for the variable in the vertical (V) screen axis.
`z`	a vector of values for the variable in the out-of-screen (O) axis.
`scaling`	the scaling applied. Two possible values are `"abc"` and `"aaa"`.
`rem.lin.trend`	a logical specifying if the linear trend should be remove. If `TRUE` then the vertical axis is replaced by e(V|H,O), i.e. the residuals from a linear fit of the vertical axis variable on the others.

uncor.vars	a logical specifying if uncorrelated H and O variables should be used. If TRUE then the O variable is replaced by e(O|H), i.e. the residuals of the regression of O on H, hence obtaining a pair of uncorrelated variables.
fit.ols	a logical specifying if a fitted OLS plane should be included.
fit.smooth	a logical specifying if a nonparametric smoothing plane should be included.
span	the span used by loess to fit the polynomial surface.
ngrid	the number of grid points to use for displaing the fitted plane.
markby	a variable (usually a factor) to be used for marking the points.
pch.points	a vector of symbols for marking the points.
col.points	a vector of colors for marking the points.
cex.points	the cex for points.
col.axis	the color of the axis.
col.ols	the color to be used for drawing the OLS plane.
col.smooth	the color to be used for drawing the smoothing plane.
background	the color of background space.
...	catch further unused arguments.

Details

This function is mainly based on the functionality of the spin-plot function once available in XLisp-Stat software https://en.wikipedia.org/wiki/XLispStat, and the adds-on introduced by the Arc software http://www.stat.umn.edu/arc/index.html.

Author(s)

Luca Scrucca <luca.scrucca@unipg.it>

References

Cook R. D., Weisberg S. (1999) *Applied Regression Including Computing and Graphics*, Wiley, Chapter 8

Examples

```
## Not run:
x1 <- rnorm(100)
x2 <- rnorm(100)
y  <- 2*x1 + x2^2 + 0.5*rnorm(100)
spinplot(x1, y, x2)
spinplot(x1, y, x2, scaling = "aaa")
spinplot(x1, y, x2, rem.lin.trend = "TRUE")
spinplot(x1, y, x2, fit.smooth = TRUE)
spinplot(x1, y, x2, fit.ols = TRUE)

x <- iris[,1:3]
y <- iris[,5]
```

```
spinplot(x)
spinplot(x, markby = y)
spinplot(x, markby = y, col.points = c("dodgerblue2", "orange", "green3"))
spinplot(x, markby = y, pch = c(0,3,1), col.points = c("dodgerblue2", "orange", "green3"))

# to save plots use
# rgl.postscript("plot.pdf", fmt="pdf")
# or
# rgl.snapshot("plot.png")

## End(Not run)
```

---

summary.msir *Summary and print methods for 'msir' objects*

Description

Summary and print methods for 'msir' objects.

Usage

```
## S3 method for class 'msir'
summary(object, numdir = object$numdir, std = FALSE, verbose = TRUE, ...)
## S3 method for class 'summary.msir'
print(x, digits = max(5, getOption("digits") - 3), ... )
```

Arguments

object	a 'msir' object
numdir	the number of directions to be shown.
std	if TRUE the coefficients basis are scaled such that all predictors have unit standard deviation.
verbose	if FALSE the coefficients basis are omitted; by default verbose = TRUE.
x	a 'summary.msir' object.
digits	the significant digits to use.
...	additional arguments.

Author(s)

Luca Scrucca <luca.scrucca@unipg.it>

See Also

msir

Index