

# Package ‘groc’

May 10, 2018

**Version** 1.0.6

**Date** 2018-05-10

**Title** Generalized Regression on Orthogonal Components

**Author** M. Bilodeau and P. Lafaye de Micheaux

**Maintainer** Pierre Lafaye de Micheaux <lafaye@unsw.edu.au>

**Imports** pls, mgcv, robust, robustbase, MASS

**Depends** rrcov

**Suggests** pppls

**Description** Robust multiple or multivariate linear regression, nonparametric regression on orthogonal components, classical or robust partial least squares models.

**License** GPL (>= 2)

**LazyLoad** yes

**NeedsCompilation** yes

**Repository** CRAN

**Date/Publication** 2018-05-10 12:16:00 UTC

## R topics documented:

corrob . . . . .	2
covrob . . . . .	3
dcov . . . . .	4
groc . . . . .	5
groc.fit . . . . .	11
grocCrossval . . . . .	13
model.frame.groc . . . . .	14
plot.groc . . . . .	15
predict.groc . . . . .	16
prim7 . . . . .	17
summary.groc . . . . .	18
<b>Index</b>	<b>20</b>

---

`corrob`*Robust correlation measure*

---

**Description**

Compute robust estimates of the correlation between two variables using the Orthogonalized Gnanadesikan-Kettenring pairwise estimator.

**Usage**

```
corrob(t, u)
```

**Arguments**

`t` a numeric vector containing the data for the first variable.  
`u` a numeric vector containing the data for the second variable.

**Details**

This function uses the `covRob` function from the **robust** package.

**Value**

Value of the robust correlation.

**Author(s)**

Martin Bilodeau (<bilodeau@dms.umontreal.ca>) and Pierre Lafaye de Micheaux (<lafaye@unsw.edu.au>)

**References**

Jiahui Wang, Ruben Zamar, Alfio Marazzi, Victor Yohai, Matias Salibian-Barrera, Ricardo Maronna, Eric Zivot, David Rocke, Doug Martin, Martin Maechler and Kjell Konis. (2013). `robust`: Robust Library. R package version 0.4-11. <https://CRAN.R-project.org/package=robust>

**See Also**

[covrob](#), [dcov](#)

**Examples**

```
data(stackloss)
corrob(stackloss$Air.Flow, stackloss$Water.Temp)
```

---

covrob	<i>Robust covariance measure</i>
--------	----------------------------------

---

### Description

Compute robust estimates of the covariance between two variables using the robust tau estimate of univariate scale, as proposed by Maronna and Zamar (2002).

### Usage

```
covrob(t, u)
```

### Arguments

t	a numeric vector containing the data for the first variable.
u	a numeric vector containing the data for the second variable.

### Details

This function uses the [scaleTau2](#) function from the **robustbase** package.

### Value

Value of the robust covariance.

### Author(s)

Martin Bilodeau (<bilodeau@dms.umontreal.ca>) and Pierre Lafaye de Micheaux (<lafaye@unsw.edu.au>)

### References

Maronna, R.A. and Zamar, R.H. (2002) Robust estimates of location and dispersion of high-dimensional datasets; *Technometrics* **44**(4), 307–317.

### See Also

[corrob](#), [dcov](#)

### Examples

```
data(stackloss)
covrob(stackloss$Air.Flow, stackloss$Water.Temp)
```

---

dcov *Distance covariance matrix.*

---

### Description

Compute the distance covariance measure of Szekely, Rizzo, and Bakirov (2007) between two samples. Warning: Only valid to compute the distance covariance for two random variables X and Y. This means that X and Y cannot be random Vectors. If this is the case, consider the package **energy**.

### Usage

```
dcov(x, y, Cpp = TRUE)
```

### Arguments

x	data of first sample
y	data of second sample
Cpp	logical. If TRUE (the default), computations are performed using a C version of the code.

### Details

See **energy**.

### Value

returns the sample distance covariance.

### Author(s)

Martin Bilodeau (<bilodeau@dms.umontreal.ca>) and Pierre Lafaye de Micheaux (<lafaye@unsw.edu.au>)

### References

Szekely, G.J., Rizzo, M.L., and Bakirov, N.K. (2007), Measuring and Testing Dependence by Correlation of Distances, *Annals of Statistics*, Vol. 35 No. 6, pp. 2769-2794.  
<http://dx.doi.org/10.1214/009053607000000505>

### See Also

[covrob](#), [corrob](#)

### Examples

```
data(stackloss)
dcov(stackloss$Air.Flow, stackloss$Water.Temp)
```

---

groc	<i>groc method</i>
------	--------------------

---

### Description

Generalized regression on orthogonal components.

### Usage

```
## Default S3 method:
groc(formula, ncomp, data, subset, na.action, plsrob =
      FALSE, method = c("lm", "lo", "s", "lts"), D = NULL,
      gamma = 0.75, Nc = 10, Ng = 20, scale = FALSE, Cpp =
      TRUE, model = TRUE, x = FALSE, y = FALSE, sp = NULL, ...)

groc(...)
```

### Arguments

formula	a model formula. Most of the <code>lm</code> formula constructs are supported. See below.
ncomp	the number of components (orthogonal components) to include in the model.
data	an optional data frame with the data to fit the model from.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
na.action	a function which indicates what should happen when the data contain missing values.
plsrob	logical. If TRUE, we use the <code>D=covrob</code> measure of dependence with the least trimmed squares <code>method="lts"</code> .
method	character giving the name of the method to use. The user can supply his own function. The methods available are linear models, "lm", local polynomials, "lo", smoothing splines, "s", and least trimmed squares, "lts".
D	function with two arguments, each one being a vector, which measures the dependence between two variables using <code>n</code> observations from them. If NULL, the covariance measure will be used. The user can supply his own function.
gamma	parameter used with the option <code>plsrob=TRUE</code> . It defines the quantile used to compute the "lts" regression. The default <code>gamma=0.75</code> gives a breakdown of 25% for a good compromise between robustness and efficiency. The value <code>gamma=0.5</code> gives the maximal breakdown of 50%.
Nc	Integer, Number of cycles in the grid algorithm.
Ng	Integer, Number of points for the grid in the grid algorithm.
scale	Logical, Should we scale the data.
Cpp	Logical, if TRUE this function will use a C++ implementation of the grid algorithm. The FALSE value should not be used, unless to get a better understanding of the grid algorithm or to compare the speed of computation between R and C++ versions of this algorithm

model	a logical. If TRUE, the model frame is returned.
x	a logical. If TRUE, the model matrix is returned.
y	a logical. If TRUE, the response is returned.
sp	A vector of smoothing parameters can be provided here. Smoothing parameters must be supplied in the order that the smooth terms appear in the model formula. Negative elements indicate that the parameter should be estimated, and hence a mixture of fixed and estimated parameters is possible. 'length(sp)' should be equal to 'ncomp' and corresponds to the number of underlying smoothing parameters.
...	further arguments to be passed to or from methods.

### Value

Y	vector or matrix of responses.
fitted.values	an array of fitted values.
residuals	residuals
T	a matrix of orthogonal components (scores). Each column corresponds to a component.
R	a matrix of directions (loadings). Each column is a direction used to obtain the corresponding component (scores).
Gobjects	contain the objects produced by the fit of the responses on the orthogonal components.
Hobjects	contain the objects produced by the "lts" fit of each deflated predictors on the orthogonal components. Hobjects are produced when plsrob=TRUE.
B	matrix of coefficients produced by the "lm" fit of each deflated predictors on the last component. B is produced when plsrob=FALSE.
Xmeans	a vector of means of the X variables.
Ymeans	a vector of means of the Y variables.
D	Dependence measure used.
V	a matrix whose columns contain the right singular vectors of the data. Computed in the preprocessing to principal component scores when the number of observations is less than the number of predictors.
dimnames	dimnames of 'fitted.values'
ncomp	the number of components used in the modelling.
method	the method used.
scale	Logical. TRUE if the responses have been scaled.
call	the function call.
terms	the model terms.
plsrob	Logical. If plsrob=TRUE, a robust partial least squares fit.
model	if model=TRUE, the model frame.

**Author(s)**

Martin Bilodeau (<bilodeau@dms.umontreal.ca>) and Pierre Lafaye de Micheaux (<lafaye@unsw.edu.au>) and Smail Mahdi (<smail.mahdi@cavehill.uwi.edu>)

**References**

Martin Bilodeau, Pierre Lafaye de Micheaux, Smail Mahdi (2015), The R Package groc for Generalized Regression on Orthogonal Components, *Journal of Statistical Software*, 65(1), 1-29, <http://www.jstatsoft.org/v65/i01/>

**Examples**

```
## Not run:
library(MASS)
#####
# Codes for Example 1 #
#####
require("groc")
data("wood")
out <- groc(y ~ x1 + x2 + x3 + x4 + x5, ncomp = 1, data = wood,
           D = corrob, method = "lts")
corrob(wood$y, fitted(out)) ^ 2
plot(out)

#####
# Codes for Example 2 #
#####
data("trees")
out <- groc(Volume ~ Height + Girth, ncomp = 1, D = spearman,
           method = "s", data = trees)
cor(trees$Volume, fitted(out)) ^ 2
plot(out$T, trees$Volume, xlab = "First component",
     ylab = "Volume", pch = 20)
lines(sort(out$T), fitted(out)[order(out$T)])
out <- boxcox(Volume ~ Height + Girth, data = trees,
             lambda = seq(-0.5, 0.5, length = 100), plotit = FALSE)
lambda <- out$x[which.max(out$y)]
out <- lm(Volume ^ lambda ~ Height + Girth, data = trees)
cor(trees$Volume, fitted(out)^(1/lambda)) ^ 2

#####
# Codes for Example 3 #
#####
data("wood")
plsr.out <- plsr(y ~ x1 + x2 + x3 + x4 + x5, data = wood)
groc.out <- groc(y ~ x1 + x2 + x3 + x4 + x5, data = wood)
apply(abs((fitted(plsr.out) - fitted(groc.out)) /
         fitted(plsr.out)), 3, max) * 100

#####
# Codes for Example 4 #
```

```
#####
set.seed(1)
n <- 200
x1 <- runif(n, -1, 1)
x2 <- runif(n, -1, 1)
y <- x1 * x2 + rnorm(n, 0, sqrt(.04))
data <- data.frame(x1 = x1, x2 = x2, y = y)
plsr.out <- plsr(y ~ x1 + x2, data = data)
groc.out <- groc(y ~ x1 + x2, D = dcov, method = "s", data = data)
plsr.v <- crossval(plsr.out, segment.type = "consecutive")
groc.v <- grocCrossval(groc.out, segment.type = "consecutive")
groc.v$validation$PRESS
plsr.v$validation$PRESS
gam.data <- data.frame(y = y, t1 = groc.out$T[, 1], t2 = groc.out$T[, 2])
gam.out <- gam(y ~ s(t1) + s(t2), data = gam.data)
par(mfrow = c(1, 2))
plot(gam.out)
par(mfrow = c(1, 1))
PRESS <- 0
for(i in 1 : 10){
  data.in <- data[-(((i - 1) * 20 + 1) : (i * 20)), ]
  data.out <- data[((i - 1) * 20 + 1) : (i * 20), ]
  ppr.out <- ppr(y ~ x1 + x2, nterms = 2, optlevel = 3, data = data.in)
  PRESS <- PRESS + sum((predict(ppr.out, newdata = data.out)-data.out$y) ^ 2)
}
PRESS

#####
# Codes for Example 5 #
#####
data("yarn")
dim(yarn$NIR)
n <- nrow(yarn)
system.time(plsr.out <- plsr(density ~ NIR, ncomp = n - 2, data = yarn))
system.time(groc.out <- groc(density ~ NIR, Nc = 20, ncomp = n - 2, data = yarn))
max(abs((fitted(plsr.out) - fitted(groc.out)) / fitted(plsr.out))) * 100
plsr.v <- crossval(plsr.out, segments = n, trace = FALSE)
plsr.v$validation$PRESS
groc.v <- grocCrossval(groc.out, segments = n, trace = FALSE)
groc.v$validation$PRESS
groc.v$validation$PREMAD

#####
# Codes for Example 6 #
#####
data("prim7")
prim7.out <- groc(X1 ~ ., ncomp = 3, D = dcov, method = "s", data = prim7)
prim7.out$R
pca <- princomp(~ ., data = as.data.frame(prim7[, -1]))
prim7.pca <- data.frame(X1 = prim7$X1, scores = pca$scores)
prim7.pca.out <- groc(X1 ~ ., ncomp = 3, D = dcov, method = "s",
                     data = prim7.pca)

pca$loadings
```



```

groc.v <- grocCrossval(prim7.out, segment.type = "consecutive")
groc.v$validation$PRESS
plsr.out <- plsr(X1 ~ ., ncomp = 3, data = prim7)
plsr.v <- crossval(plsr.out, segment.type = "consecutive")
plsr.v$validation$PRESS
PRESS <- 0
for(i in 1 : 10){
  data.in <- prim7[-(((i - 1) * 50 + 1) : (i * 50)), ]
  data.out <- prim7[(((i - 1) * 50 + 1) : (i * 50)), ]
  ppr.out <- ppr(X1 ~ ., nterms = 3, optlevel = 3, data = data.in)
  PRESS <- PRESS + sum((predict(ppr.out, newdata = data.out) - data.out$X1) ^ 2)
}
PRESS

#####
# Codes for Example 7 #
#####
n <- 50 ; B <- 30
mat.cor <- matrix(0, nrow = B, ncol = 3) ; mat.time <- matrix(0, nrow = B, ncol = 3)
for (i in 1:B) {
  X <- matrix(runif(n * 5, -1, 1), ncol = 5)
  A <- matrix(runif(n * 50, -1, 1), nrow = 5)
  y <- (X[,1] + X[,2])^2 + (X[,1] + 5 * X[,2])^2 + rnorm(n)
  X <- cbind(X, X)
  D <- data.frame(X = X, y = y)
  mat.time[i,1] <- system.time(out1 <- plsr(y ~ X, , ncomp = 2, data = D))[1]
  mat.time[i,2] <- system.time(out2 <- ppr(y ~ X, , nterms = 2, data = D))[1]
  mat.time[i,3] <- system.time(out3 <- groc(y ~ X, D = dcov, method = "s", ncomp = 2, data = D))[1]
  mat.cor[i,] <- cor(y, cbind(fitted(out1)[,2], fitted(out2), fitted(out3)[,2]))
}
colMeans(mat.cor)
colMeans(mat.time)

#####
# Codes for Example 8 #
#####
data("oliveoil")
n <- nrow(oliveoil)
plsr.out <- plsr(sensory ~ chemical, data = oliveoil, method = "simpls")
groc.out <- groc(sensory ~ chemical, data = oliveoil)
max(abs((fitted(plsr.out) - fitted(groc.out)) / fitted(plsr.out))) * 100
groc.v <- grocCrossval(groc.out, segments = n)
groc.v$validation$PRESS
colMeans(groc.v$validation$PRESS)
Y <- oliveoil$sensory
for (j in 1 : ncol(Y)) print(cor(Y[, j], fitted(groc.out)[, j, 2]))

#####
# Codes for Example 9 #
#####
require("ppls")
data("cookie")
X <- as.matrix(log(cookie[1 : 40, 51 : 65]))

```

```

Y <- as.matrix(cookie[1 : 40, 701 : 704])
X <- X[, 2 : 601] - X[, 1 : 600]
data <- data.frame(Y = I(Y), X = I(X))
n <- nrow(data)
q <- ncol(Y)
x1 <- "Wavelength index"
y1 <- "First differences of log(1/reflectance)"
matplot(1:ncol(X), t(X), lty = 1, xlab = x1, ylab = y1, type = "l")
out1 <- plsr(Y ~ X, ncomp = n - 2, data = data)
cv <- crossval(out1, segments = n)
cv.mean <- colMeans(cv$validation$PRESS)
plot(cv.mean, xlab = "h", ylab = "Average PRESS", pch = 20)
h <- 3
for (j in 1 : q) print(cor(Y[, j], fitted(out1)[, j, h]))
set.seed(1)
out2 <- groc(Y ~ X, ncomp = h, data = data, plsrob = TRUE)
for (j in 1 : q) print(corrob(Y[, j], fitted(out2)[, j, h]))
plot(out2)

#####
# Codes for Example 10 #
#####
set.seed(2)
n <- 30
t1 <- sort(runif(n, -1, 1))
y <- t1 + rnorm(n, mean = 0, sd = .05)
y[c(14, 15, 16)] <- y[c(14, 15, 16)] + .5
data <- data.frame(x1 = t1, x2 = 2 * t1, x3 = -1.5 * t1, y = y)
out <- groc(y ~ x1 + x2 + x3, ncomp = 1, data = data, plsrob = TRUE)
tau <- scaleTau2(residuals(out), mu.too = TRUE)
std.res <- scale(residuals(out), center = tau[1], scale = tau[2])
index <- which(abs(std.res)>3)
prm.res <- read.table("prmresid.txt")
plot(t1, y, pch = 20)
matlines(t1, cbind(t1,fitted(out), y - prm.res), lty = 1 : 3)
legend(.4, -.5, legend = c("true model", "groc", "prm"), lty = 1 : 3)
text(t1[index], y[index], index, cex = .8, pos = 3)

#####
# Codes for Example 11 #
#####
data("pulpfiber")
X <- as.matrix(pulpfiber[, 1:4])
Y <- as.matrix(pulpfiber[, 5:8])
data <- data.frame(X = I(X), Y = I(Y))
set.seed(55481)
out.rob <- groc(Y ~ X, data = data, plsrob = TRUE)
plot(out.rob, cex = .6)
out.simpls <- groc(Y ~ X, data = data)
cv.rob <- grocCrossval(out.rob, segment.type = "consecutive")
PREMAD.rob <- cv.rob$validation$PREMAD[,4]
PREMAD.rob
cv.simpls <- grocCrossval(out.simpls, segment.type = "consecutive")

```

```
PREMAD.simpls <- cv.simpls$validation$PREMAD[,4]
PREMAD.simpls
(PREMAD.rob - PREMAD.simpls) / PREMAD.simpls * 100

## End(Not run)
```

groc.fit

*Fitting a groc model***Description**

Fits a groc model with the grid algorithm.

**Usage**

```
groc.fit(X, Y, ncomp = min(nrow(X) - 1, ncol(X)), D = NULL, gamma =
  0.75, method = NULL, plsrob = FALSE, Nc = 10, Ng = 20,
  scale = FALSE, Cpp = TRUE, stripped = FALSE, maxiter =
  100, sp = NULL, ...)
```

**Arguments**

X	a matrix of predictors. NAs and Infs are not allowed.
Y	a vector or matrix of responses. NAs and Infs are not allowed.
ncomp	the number of components to be used in the modelling.
D	Dependence measure.
gamma	Used to set the breakdown value when method="lts".
method	the method to be used. Currently only 'lm', 'lo', 's', and 'lts'.
plsrob	Logical. If TRUE, the function sets D=covrov and method="lts" for a robust partial least squares fit.
Nc	Integer. Number of cycles in the grid algorithm
Ng	Integer. Number of points for the grid in the grid algorithm.
scale	Logical. If TRUE the responses are scaled.
Cpp	Logical. If TRUE, computations are performed in a faster way using a C code.
stripped	logical. If TRUE the calculations are stripped as much as possible for speed; this is meant for use with cross-validation or simulations when only the coefficients are needed. Defaults to FALSE.
maxiter	Integer. Maximal number of iterations in the grid algorithm. Used only when there are more than one response.

sp A vector of smoothing parameters can be provided here. Smoothing parameters must be supplied in the order that the smooth terms appear in the model formula. Negative elements indicate that the parameter should be estimated, and hence a mixture of fixed and estimated parameters is possible. 'length(sp)' should be equal to 'ncomp' and corresponds to the number of underlying smoothing parameters.

... other arguments. Currently ignored.

### Value

Y data used as response.

fitted.values an array of fitted values. Its element [i,j,k] is the fitted value for observation i, response j, and when k components are used.

residuals an array of regression residuals. It has the same dimensions as fitted.values.

T a matrix of orthogonal components (scores). Each column corresponds to a component.

R a matrix of directions (loadings). Each column is a direction used to obtain the corresponding component (scores).

Gobjects contain the objects produced by the fit of the responses on the orthogonal components.

Hobjects contain the objects produced by the "lts" fit of each deflated predictors on the orthogonal components. Hobjects are produced when plsrob=TRUE.

B matrix of coefficients produced by the "lm" fit of each deflated predictors on the last component. B is produced when plsrob=FALSE.

Xmeans a vector of means of the X variables.

Ymeans a vector of means of the Y variables.

D Dependence measure used.

V a matrix whose columns contain the right singular vectors of the data. Computed in the preprocessing to principal component scores when the number of observations is less than the number of predictors.

dimnames dimnames of 'fitted.values'

### Author(s)

Martin Bilodeau (<bilodeau@dms.umontreal.ca>) and Pierre Lafaye de Micheaux (<lafaye@unsw.edu.au>)

### References

Martin Bilodeau, Pierre Lafaye de Micheaux, Smail Mahdi (2015), The R Package groc for Generalized Regression on Orthogonal Components, *Journal of Statistical Software*, 65(1), 1-29, <http://www.jstatsoft.org/v65/i01/>

---

grocCrossval                      *Cross-validation of groc models*

---

### Description

A “stand alone” cross-validation function for groc objects.

### Usage

```
grocCrossval(object, segments = 10, segment.type = c("random",
  "consecutive", "interleaved"), length.seg, trace = 15, ...)
```

### Arguments

object	a groc object; the regression to cross-validate.
segments	the number of segments to use, or a list with segments (see below).
segment.type	the type of segments to use.
length.seg	Positive integer. The length of the segments to use.
trace	if TRUE, tracing is turned on. If numeric, it denotes a time limit (in seconds). If the estimated total time of the cross-validation exceeds this limit, tracing is turned on.
...	additional arguments, sent to the underlying fit function.

### Details

This function performs cross-validation on a model fit by groc. It can handle models such as `groc(Y ~ X, ...)`.

Note that to use `grocCrossval`, the data *must* be specified with a data argument when fitting object.

If `segments` is a list, the arguments `segment.type` and `length.seg` are ignored. The elements of the list should be integer vectors specifying the indices of the segments.

Otherwise, segments of type `segment.type` are generated. How many segments to generate is selected by specifying the number of segments in `segments`, or giving the segment length in `length.seg`. If both are specified, `segments` is ignored.

When tracing is turned on, the segment number is printed for each segment.

### Value

The supplied object is returned, with an additional component `validation`, which is a list with components

method	equals "CV" for cross-validation.
pred	an array with the cross-validated predictions.
PRESS	a matrix of PRESS values for models with 1, ..., <code>ncomp</code> components. Each row corresponds to one response variable.

PREMAD	a matrix of PREMAD values for models with 1, ..., ncomp components. Each row corresponds to one response variable.
RMSEP	a matrix of $\sqrt{\text{PRESS}/\text{nobj}}$ values for models with 1, ..., ncomp components. Each row corresponds to one response variable.
segments	the list of segments used in the cross-validation.
ncomp	the number of components.

**Author(s)**

Martin Bilodeau (<bilodeau@dms.umontreal.ca>) and Pierre Lafaye de Micheaux (<lafaye@unsw.edu.au>)

**References**

Martin Bilodeau, Pierre Lafaye de Micheaux, Smail Mahdi (2015), The R Package groc for Generalized Regression on Orthogonal Components, *Journal of Statistical Software*, 65(1), 1-29, <http://www.jstatsoft.org/v65/i01/>

**Examples**

```
data(yarn,package="pls")
yarn.groc <- groc(density ~ NIR, 6, data = yarn)
yarn.cv <- grocCrossval(yarn.groc, segments = 10)

yarn.cv$validation$PRESS
yarn.cv$validation$PREMAD
```

---

model.frame.groc      *Extract Information From a Fitted groc Model*

---

**Description**

Functions to extract information from groc objects: the model frame, the model matrix.

**Usage**

```
## S3 method for class 'groc'
model.matrix(object, ...)
## S3 method for class 'groc'
model.frame(formula, ...)
```

**Arguments**

object, formula      a groc object. The fitted model.

...                    other arguments sent to underlying functions.

**Details**

`model.frame.groc` returns the model frame; i.e. a data frame with all variables necessary to generate the model matrix. See [model.frame](#) for details.

`model.matrix.groc` returns the (possibly coded) matrix used as  $X$  in the fitting. See [model.matrix](#) for details.

**Value**

`model.frame.groc` returns a data frame with all variables necessary to generate the model matrix.

`model.matrix.groc` returns the  $X$  matrix.

**Author(s)**

Ron Wehrens and Bjørn-Helge Mevik

**See Also**

[coef](#), [fitted](#), [residuals](#), [model.frame](#)

---

plot.groc

*Plot groc objects.*

---

**Description**

A function to plot groc objects.

**Usage**

```
## S3 method for class 'groc'
plot(x, h=x$ncomp, cex=0.8, ...)
```

**Arguments**

<code>x</code>	A groc object.
<code>h</code>	Number of components in the model.
<code>cex</code>	Character expansion factor for point labels.
<code>...</code>	Further arguments passed to internal plot function.

**Details**

If `plsrob=FALSE`, a plot of robust Mahalanobis distances for residuals versus robust Mahalanobis distances for components. Useful for identification of good points, vertical outliers, good and bad leverage points.

If `plsrob=TRUE`, the previous plot is done with another similar plot of classical Mahalanobis distances to compare the identification of the various type of points obtained by classical or robust partial least squares.

**Author(s)**

Martin Bilodeau (<bilodeau@dms.umontreal.ca>) and Pierre Lafaye de Micheaux (<lafaye@unsw.edu.au>)

**References**

Martin Bilodeau, Pierre Lafaye de Micheaux, Smail Mahdi (2015), The R Package groc for Generalized Regression on Orthogonal Components, *Journal of Statistical Software*, 65(1), 1-29, <http://www.jstatsoft.org/v65/i01/>

**Examples**

```
## This example takes some time:
## Not run:
data("pulpfiber", package="robustbase")
X <- as.matrix(pulpfiber[, 1:4])
Y <- as.matrix(pulpfiber[, 5:8])
data <- data.frame(X=I(X), Y=I(Y))
set.seed(55481)
out.rob <- groc(Y ~ X, data=data, plsrob=TRUE)
plot(out.rob, cex=.6)

## End(Not run)
```

---

predict.groc

*Predict Method for groc*

---

**Description**

Prediction for groc models. New responses or scores are predicted using a fitted model and a new matrix of observations.

**Usage**

```
## S3 method for class 'groc'
predict(object, newdata, ncomp = object$ncomp, na.action = na.pass, ...)
```

**Arguments**

object	a groc object. The fitted model
newdata	a data frame. The new data. If missing, the training data is used.
ncomp	vector of positive integers. The components to use in the prediction.
na.action	function determining what should be done with missing values in newdata. By default, nothing is done.
...	further arguments. Currently not used



**Value**

A three dimensional array of predicted response values is returned. The dimensions correspond to the observations, the response variables and the model sizes, respectively.

**Author(s)**

Martin Bilodeau (<bilodeau@dms.umontreal.ca>) and Pierre Lafaye de Micheaux (<lafaye@unsw.edu.au>)

**References**

Martin Bilodeau, Pierre Lafaye de Micheaux, Smail Mahdi (2015), The R Package groc for Generalized Regression on Orthogonal Components, *Journal of Statistical Software*, 65(1), 1-29, <http://www.jstatsoft.org/v65/i01/>

**See Also**

[plot.groc](#)

**Examples**

```
data("wood",package="robustbase")
out <- groc(y ~ x1+x2+x3+x4+x5, ncomp=1, data=wood,D=corrob, method="lts")
predict(out)

newdata<- data.frame(x1= 0.5, x2=0.1, x3=0.4, x4=0.5, x5=0.8)
predict(out,newdata)
```

---

prim7

*prim7 Dataset*

---

**Description**

The data prim7 is a particle physics experiment analyzed by projection pursuit regression in Friedman and Stuetzle (1981). It has 7 variables on 500 observations. The data set is described in Friedman and Tukey (1974).

**Format**

This data frame contains the following columns:

- X1** First variable.
- X2** Second variable.
- X3** Third variable.
- X4** Fourth variable.
- X5** Fifth variable.
- X6** Sixth variable.
- X7** Seventh variable.

**References**

Friedman and Tukey (1974), A Projection Pursuit Algorithm for Exploratory Data Analysis, *IEEE Transactions on Computers* (Volume:C-23, Issue: 9)

Friedman, Jerome H.; Stuetzle, Werner (1981), Projection pursuit regression. *J. Amer. Statist. Assoc.* 76, no. 376, 817–823.

**Examples**

```
data(prim7)
```

---

```
summary.groc
```

```
Summary and Print Methods for groc objects
```

---

**Description**

Summary and print methods for groc objects.

**Usage**

```
## S3 method for class 'groc'
summary(object, what = "validation",
        digits = 4, print.gap = 2, ...)
## S3 method for class 'groc'
print(x, ...)
```

**Arguments**

x, object	a groc object
what	character, only "validation" for the moment
digits	integer. Minimum number of significant digits in the output. Default is 4.
print.gap	Integer. Gap between coloumns of the printed tables.
...	Other arguments sent to underlying methods.

**Details**

If what is "validation", the cross-validated PRESS, RPEMAD and RMSEPs (if available) are given.

**Value**

print.groc return the object invisibly.

**Author(s)**

P. Lafaye de Micheaux

## References

Martin Bilodeau, Pierre Lafaye de Micheaux, Smail Mahdi (2015), The R Package groc for Generalized Regression on Orthogonal Components, *Journal of Statistical Software*, 65(1), 1-29, <http://www.jstatsoft.org/v65/i01/>

## See Also

[groc](#), [grocCrossval](#)

## Examples

```
data("yarn", package="pls")
yarn.groc <- groc(density ~ NIR, 6, data = yarn)
yarn.cv <- grocCrossval(yarn.groc, segments = 10)
print(yarn.groc)
summary(yarn.cv)
```

# Index

- \*Topic **datasets**
  - prim7, [17](#)
- \*Topic **distribution**
  - groc, [5](#)
- \*Topic **htest**
  - groc, [5](#)
- \*Topic **multivariate**
  - dcov, [4](#)
  - groc.fit, [11](#)
  - grocCrossval, [13](#)
  - model.frame.groc, [14](#)
  - plot.groc, [15](#)
  - predict.groc, [16](#)
  - summary.groc, [18](#)
- \*Topic **regresion**
  - plot.groc, [15](#)
- \*Topic **regression**
  - groc.fit, [11](#)
  - grocCrossval, [13](#)
  - model.frame.groc, [14](#)
  - predict.groc, [16](#)
  - summary.groc, [18](#)
- \*Topic **robust**
  - corrob, [2](#)
  - covrob, [3](#)

coef, [15](#)  
corrob, [2](#), [3](#), [4](#)  
covRob, [2](#)  
covrob, [2](#), [3](#), [4](#)

dcov, [2](#), [3](#), [4](#)

fitted, [15](#)

groc, [5](#), [19](#)  
groc.fit, [11](#)  
grocCrossval, [13](#), [19](#)

model.frame, [15](#)  
model.frame.groc, [14](#)

model.matrix, [15](#)  
model.matrix.groc (model.frame.groc), [14](#)

plot.groc, [15](#), [17](#)  
predict.groc, [16](#)  
prim7, [17](#)  
print.groc (summary.groc), [18](#)

residuals, [15](#)

scaleTau2, [3](#)  
summary.groc, [18](#)