

# Package ‘grt’

May 8, 2026

**Type** Package

**Title** General Recognition Theory

**Version** 0.2.1

**Date** 2017-07-14

**Author** Kazunaga Matsuki

**Maintainer** Andy Wills <andy@willslab.co.uk>

**Description** Functions to generate and analyze data for psychology experiments based on the General Recognition Theory.

**License** GPL (>= 2)

**Depends** R (>= 3.2.0), grDevices, graphics, stats, utils, MASS

**Suggests** rgl, misc3d

**LazyLoad** yes

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2017-07-14 17:40:11 UTC

## Contents

grt-package	2
coef.glc	3
dprime	4
extractAIC	5
gaborPatch	7
gcjc	9
gcjcStruct	10
glc	11
glcStruct	13
gqc	14
gqcStruct	16
grg	16
grtMeans	18

grtrnorm . . . . .	19
ldb . . . . .	20
ldb.p.correct . . . . .	22
lines.gqcStruct . . . . .	23
logLik.glc . . . . .	24
logLik.glcStruct . . . . .	25
logLik.gqc . . . . .	26
logLik.gqcStruct . . . . .	26
mcovs . . . . .	27
new2old_par . . . . .	28
old2new_par . . . . .	29
plot.gcjc . . . . .	30
plot.glc . . . . .	31
plot.gqc . . . . .	32
plot3d.glc . . . . .	33
plot3d.gqc . . . . .	34
predict.glc . . . . .	36
qdb . . . . .	37
qdb.p.correct . . . . .	38
scale . . . . .	39
subdemo_1d . . . . .	40
subdemo_2d . . . . .	41
subdemo_3d . . . . .	42
subdemo_cj . . . . .	42
unscale . . . . .	43
<b>Index</b>	<b>45</b>

---

grt-package

*General Recognition Theory*


---

## Description

Functions to generate and analyze data for psychology experiments based on the General Recognition Theory.

## Details

This package is written based mostly on the GRT Toolbox for MATLAB by Alfonso-Reese (2006), although many functions have been renamed and modified from the original in order to make them more general and “R-like.”

The functions `grtrnorm` and `grtMeans` are used for design categorization experiments and generating stimuli. The functions `glc`, `gcjc`, `gqc`, and `grg` are used for fitting the general linear classifier, the general conjunctive classifier, the general quadratic classifier, and the general random guessing model, respectively. The `glc`, `gcjc`, and `gqc` have plot methods (`plot.glc`, `plot.gcjc`, `plot.gqc`, `plot3d.glc`, `plot3d.gqc`).

For a complete list of functions, use `library(help = "catlearn")`.

**Author(s)**

Kazunaga Matsuki

Maintainer: Andy Wills &lt;andy@willslab.co.uk&gt;

**References**

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

Ashby, F. G., & Gott, R. E. (1988). *Decision rules in the perception and categorization of multidimensional stimuli*. Journal of Experimental Psychology: Learning, Memory, & Cognition, 14, 33-53.

Ashby, F. G. (1992) *Multidimensional models of perception and cognition*. Lawrence Erlbaum Associates.

---

coef.glc	<i>Extract 'glc' or 'gcjc' coefficients</i>
----------	---

---

**Description**

Extracts the coefficients from the model object glc, glcStruct, or gcjc.

**Usage**

```
## S3 method for class 'glc'
coef(object, ...)

## S3 method for class 'glcStruct'
coef(object, ...)

## S3 method for class 'gcjc'
coef(object, ...)
```

**Arguments**

object	object of class glc or glcStruct
...	further arguments

**Details**

Both the object glc and glcStruct contain the parameters for the decision boundary in the form:

$$a_1x_1 + a_2x_2 \dots a_nx_n + b = 0$$

This function transforms and returns the coefficients of the function solved with respect the  $x_n$ .

For the object gcjc, a list of two coefficients (Intercepts) are returned.

## Examples

```

data(subjdemo_2d)
fit.2dl <- glc(response ~ x + y, data=subjdemo_2d,
  category=subjdemo_2d$category, zlimit=7)
plot(fit.2dl, fitdb=FALSE)
abline(coef(fit.2dl), col = "red")
abline(coef(fit.2dl$initpar))

fit.1dx <- update(fit.2dl, . ~ . -y)
abline(v=coef(fit.1dx), col="green")

fit.1dy <- update(fit.2dl, . ~ . -x)
abline(h=coef(fit.1dy), col="blue")

```

---

dprime	<i>Calculate d' (d-prime)</i>
--------	-------------------------------

---

## Description

Obtain the standardized distance between the two probability distributions, known as  $d'$  or sensitivity index.

## Usage

```

dprime(x,
  category,
  response,
  par = list(),
  zlimit = Inf,
  type = c("SampleIdeal", "Observer"))

dprimef(means, covs, noise=NULL)

```

## Arguments

x	a data frame or matrix containing samples from two multivariate normal distributions.
category	a vector or factor of labels of populations to which the samples belong
response	a vector or factor specifying the participant's classification responses for each samples
par	object of class <code>glcStruct</code> or a named list containing a set of parameters that specify a general linear decision bound. The list should contain <code>noise</code> , <code>coeffs</code> , <code>bias</code> .
zlimit	numeric. The z-scores (or discriminant scores) beyond the specified value will be truncated and replaced with that value. Default to <code>Inf</code>

type	a character string specifying the type of $d'$ to be returned. If <code>SampleIdeal</code> , $d'$ is calculated based on ideal (or true) category membership as specified in <code>category</code> . If <code>Observer</code> , $d'$ is calculated using the response vector as a grouping factor.
means	a list of numeric vectors containing the means of two distributions
covs	a matrix or a list of matrices containing the variance-covariance matrix of the two distributions
noise	numeric. perceptual and criterial noise expressed as standard deviation. Default to NULL

### Details

The function `dprime` estimates  $d'$  from sample data sets, whereas the function `dprimef` calculates it from population parameters.

In `dprime`, if any parts of the argument `par` are missing, the function will estimate an optimal linear decision bound from supplied `x` and `category`. The argument `response` is not used if `type` is `SampleIdeal`.

### Author(s)

Author of the original Matlab routines: Leola Alfonso-Reese

Author of R adaptation: Kazunaga Matsuki

### References

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

### Examples

```
data(subjdemo_2d)
d2 <- subjdemo_2d
db <- glcStruct(noise=10, coeffs=c(0.514,-0.857),bias=-0.000154)
dprime(d2[,2:3], d2$category, d2$response, par = db, zlimit=7, type='SampleIdeal')

mc <- mcovs(category ~ x + y, data=d2)
dprimef(mc$means, mc$covs)
```

---

extractAIC

*extractAIC method for class 'glc', 'gqc', 'gjc', and 'grg'*

---

### Description

Extract Akaike's An Information Criteria from a General Linear, Quadratic, or Conjunctive Classifier, or a General Random Guessing model

**Usage**

```
## S3 method for class 'glc'
extractAIC(fit, scale, k = 2, ...)

## S3 method for class 'gqc'
extractAIC(fit, scale, k = 2, ...)

## S3 method for class 'gcjc'
extractAIC(fit, scale, k = 2, ...)

## S3 method for class 'grg'
extractAIC(fit, scale, k = 2, ...)
```

**Arguments**

fit	object of class glc, gqc, gcjc, or grg
scale	unused argument
k	numeric specifying the penalty per parameter to be used in calculating AIC. Default to 2.
...	further arguments (currently not used).

**Details**

As with the default method, the criterion used is

$$AIC = -2\log L + k \times df,$$

where  $L$  is the likelihood and  $df$  is the degrees of freedom (i.e., the number of free parameters) of fit.

**Value**

A numeric vector of length 2 including:

df	the degrees of freedom for the fitted model fit.
AIC	the Akaike's Information Criterion for fit.

**Examples**

```
data(subjdemo_2d)
#fit a 2d suboptimal model
fit.2dl <- glc(response ~ x + y, data=subjdemo_2d,
  category=subjdemo_2d$category, zlimit=7)
extractAIC(fit.2dl)
```

---

`gaborPatch`*Draw a gray-scale Gabor Patch*

---

**Description**

Draw a gray-scale Gabor Patch

**Usage**

```
gaborPatch(sf,  
  theta = 0,  
  rad = (theta * pi)/180,  
  pc = 1,  
  sigma = 1/6,  
  psi = 0,  
  grating = c("cosine", "sine"),  
  npoints = 100,  
  trim = 0,  
  trim.col = .5,  
  ...)
```

**Arguments**

<code>sf</code>	number of cycles per image.
<code>theta</code>	orientation in degree. See ‘Details’
<code>rad</code>	orientation in radian
<code>pc</code>	a fraction (0 to 1) specifying the peak contrast of the Gabor
<code>sigma</code>	the standard deviation of the Gaussian mask. Either a single numeric or a numeric vector of length 2.
<code>psi</code>	phase offset in radian
<code>grating</code>	type of grating to be used. Default to ‘cosine’.
<code>npoints</code>	number of points per line used to draw the patch.
<code>trim</code>	Gaussian values smaller than the specified value should be trimmed.
<code>trim.col</code>	gray level of the trimmed part of the image, between 0 (‘black’) and 1 (‘white’). Default to .5 (‘gray’) Setting it to any other value or NA makes the trimmed part transparent.
<code>...</code>	additional parameters for <a href="#">image</a> may be passed as arguments to this function.

**Details**

The arguments `theta` and `rad` is the same thing but in different units. If both are supplied, `rad` takes the precedence.

**Value**

invisibly returns the matrix of the plotted values.

**Note**

This function is written just for fun; it is not optimized for speed or for performance.

**References**

Fredericksen, R. E., Bex, P. J., & Verstraten, F. A. J. (1997). *How Big is a Gabor and Why Should We Care?* Journal of the Optical Society of America A. 14, 1–12.

Gabor Filter. (2010, June 5). In *Wikipedia, the free encyclopedia*. Retrieved July 7, 2010, from [http://en.wikipedia.org/wiki/Gabor\\_filter](http://en.wikipedia.org/wiki/Gabor_filter)

**Examples**

```
# An imitation of Fredericksen et al.'s (1997) Fig 1.
# that demonstrate the relation between peak contrast
# and perceived size of the Gabor

op <- par(mfcol = c(3, 3), pty = "m", mai = c(0,0,0,0))
for(i in c(.85, .21, .06)){
  for(j in c(1/6, 1/7, 1/8)){
    gaborPatch(20, pc = i, sigma = j)
  }
}
par(op)

## Not run:
# a typical plot of the stimuli and category structure
# often seen in artificial category-learning literatures.
m <- list(c(268, 157), c(332, 93))
covs <- matrix(c(4538, 4351, 4351, 4538), ncol = 2)
II <- grtrnorm(n = 40, np = 2, means = m, covs = covs,
  clip.sd = 4, seed = 1234)
II$sf <- .25+(II$x1/50)
II$theta <- II$x2*(18/50)

plot(II[,2:3], xlim = c(-100,600), ylim = c(-200,500),
  pch = 21, bg = c("white","gray")[II$category])
abline(a = -175, b = 1)

library(Hmisc)
idx <- c(20, 31, 35, 49, 62)
xpos <- c(0, 100, 300, 350, 550)
ypos <- c(50, 300, 420, -120, 50)
for(i in 1:5)
{
  j = idx[i]
  segments(x0=II[j,"x1"], y0=II[j,"x2"], x1=xpos[i], y1=ypos[i])
  subplot(gaborPatch(sf=II[j,"sf"], theta=II[j,"theta"]), x=xpos[i], y=ypos[i])
}
```

```

}
## End(Not run)

```

---

gcjc

*General Conjunctive Classifier*


---

## Description

Fit a general conjunctive classifier.

## Usage

```

gcjc(formula, data, category, par, config = 1, zlimit = Inf,
      fixed = list(), equal.noise = TRUE, opt = c("nlminb", "optim"),
      lower=-Inf, upper=Inf, control=list())

```

## Arguments

formula	A formula of the form $\text{response} \sim x_1 + x_2 + \dots$ , where the response specifies the grouping factor (generally a participant's response) and the right hand side specifies the relevant dimensions or features of the stimuli.
data	A data frame from which variables specified in formula are taken.
category	(Optional.) A factor specifying the true category membership of the stimuli.
par	object of class <code>gcjcStruct</code> or a named list containing a set of initial parameters - that is, noise and bias (intercepts).
config	A numeric value specifying the location of the conjunctive category in relation to the category bounds. The value 1 indicates the category is on the top right (set as default), 2 indicates the top left, 3 indicates bottom left, and 4 indicates the bottom right.
zlimit	numeric. The z-scores (or discriminant scores) beyond the specified value will be truncated. Default to <code>Inf</code> .
fixed	A named list of logical vectors specifying whether each of noise and bias parameters should be fixed to the initial value. Default to <code>list(noise=c(FALSE, TRUE), bias=FALSE)</code> . A fatal error will be returned if set to all TRUE.
equal.noise	logical. If set to TRUE (default), two bounds will have the same noise parameter.
opt	A character string specifying the optimizer to be used: either <code>nlminb</code> (the default) or <code>optim</code> . If "optim", "L-BFGS-B" method is used (see 'Details' of <code>optim</code> ).
lower, upper	Bounds on the parameters. see 'Details' for default values.
control	A list of control parameters passed to the internal optimization function. See 'Details' of <code>nlminb</code> or <code>optim</code> .

**Details**

If `par` is not fully specified in the argument, the function attempts to calculate the initial parameter values based on means by category or by response.

The default lower and upper values are selected based on the range of the data input so that the decision bound is found within the range of the data and convergence can be reached.

**Value**

object of the class `gcjc`, i.e., a list containing the following components:

<code>terms</code>	the <code>terms</code> object used.
<code>call</code>	the matched call.
<code>model</code>	the design matrix used to fit the model.
<code>category</code>	the category vector as specified in the input.
<code>initpar</code>	the initial parameter used to fit the model.
<code>par</code>	the fitted parameter.
<code>logLik</code>	the log-likelihood at convergence.

**References**

Ashby, F. G. (1992) *Multidimensional models of perception and cognition*. Lawrence Erlbaum Associates.

**See Also**

[glc](#), [logLik.gcjc](#), [coef.gcjc](#), [plot.gcjc](#)

**Examples**

```
data(subjdemo_cj)

m.cj <- gcjc(response ~ x1 + x2, data=subjdemo_cj,
             config=2, category=subjdemo_cj$category, zlimit=7)
```

---

gcjcStruct

*General Conjunctive Classifier structure*

---

**Description**

A list of model parameters that specify a conjunctive decision bound, containing noise, coeffs, and bias.

**Usage**

```
gcjcStruct(noise, bias, config=c(1,2,3,4))
```

**Arguments**

noise	a positive non-zero numeric.
bias	numeric vector corresponding to the intercepts of the bounds
config	A numeric value specifying the location of the conjunctive category in relation to the category bounds. The value 1 indicates the category is on the top right (set as default), 2 indicates the top left, 3 indicates bottom left, and 4 indicates the bottom right.

**Value**

object of class `gcjcStruct`, which is a list of a named list containing `noise`, `coeffs`, and `bias`.

**See Also**

[gcjc](#), [coef.glcStruct](#), [logLik.glcStruct](#)

**Examples**

```
params <- gcjcStruct(noise=10, bias=c(100, 200), config=1)
```

---

glc

*General Linear Classifier*


---

**Description**

Fit a general linear classifier (a.k.a. linear decision-bound model).

**Usage**

```
glc(formula, data, category, par = list(), zlimit = Inf,
     covstruct=c("unstructured", "scaledIdentity", "diagonal", "identity"),
     fixed = list(), opt = c("nlminb", "optim"),
     lower=-Inf, upper=Inf, control=list())
```

**Arguments**

formula	A formula of the form <code>response ~ x1 + x2 + ...</code> , where the response specifies the grouping factor (generally a participant's response) and the right hand side specifies the relevant dimensions or features of the stimuli.
data	A data frame from which variables specified in <code>formula</code> are taken.
category	(Optional.) A factor specifying the true category membership of the stimuli.
par	object of class <code>glcStruct</code> or named list containing a set of initial parameters (i.e., <code>noise</code> , <code>coeffs</code> , <code>bias</code> ) used to fit the data.
zlimit	numeric. The z-scores (or discriminant scores) beyond the specified value will be truncated. Default to <code>Inf</code> .

covstruct	An optional character string. Only used when the initial parameters are not fully specified. see <a href="#">ldb</a> .
fixed	A named list of logical vectors specifying whether each of noise, coeffs, and bias parameters should be fixed to the initial value. Default to <code>list(noise=FALSE, coeffs=FALSE, bias=FALSE)</code> . A fatal error will be returned if set to all TRUE.
opt	A character string specifying the optimizer to be used: either <a href="#">nlminb</a> (the default) or <a href="#">optim</a> . If “optim”, “L-BFGS-B” method is used (see ‘Details’ of <a href="#">optim</a> ).
lower, upper	Bounds on the parameters. see ‘Details’ for default values.
control	A list of control parameters passed to the internal optimization function. See ‘Details’ of <a href="#">nlminb</a> or <a href="#">optim</a> .

### Details

If `par` is not fully specified in the argument, the function attempts to calculate the initial parameter values by internally calling the functions `mcovs` and `ldb`. If `category` is also not specified, the response specified in the `formula` is used as the grouping factor in `mcovs`.

The default lower and upper values vary depending on the dimension of the model (i.e., the number of variables in the right hand side of `formula`). In all cases, default lower and upper values for the noise parameter is .001 and 500 respectively. In cases when an one-dimensional model is fitted, lower and upper bounds for the bias parameters are selected based on the range of the data input so that the decision bound is found within the reasonable range of the data and convergence can be reached. In all other cases, `coeffs` and `bias` has no limits.

When an one-dimensional model is being fit, `fixed$coeffs` always becomes TRUE.

### Value

object of the class `glc`, i.e., a list containing the following components:

<code>terms</code>	the <a href="#">terms</a> object used.
<code>call</code>	the matched call.
<code>model</code>	the design matrix used to fit the model.
<code>category</code>	the category vector as specified in the input.
<code>initpar</code>	the initial parameter used to fit the model.
<code>par</code>	the fitted parameter.
<code>logLik</code>	the log-likelihood at convergence.

### References

- Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.
- Ashby, F. G., & Gott, R. E. (1988). *Decision rules in the perception and categorization of multidimensional stimuli*. Journal of Experimental Psychology: Learning, Memory, & Cognition, 14, 33-53.
- Ashby, F. G. (1992) *Multidimensional models of perception and cognition*. Lawrence Erlbaum Associates.

**See Also**

[gqc](#), [ldb](#), [logLik.glc](#), [coef.glc](#), [predict.glc](#), [scale.glc](#), [plot.glc](#), [plot3d.glc](#)

**Examples**

```
data(subjdemo_2d)
d2 <- subjdemo_2d
#fit a 2d suboptimal model
fit.2dl <- glc(response ~ x + y, data=d2, category=d2$category, zlimit=7)
#fit a 1d model (on the dimension 'y') on the same dataset
fit.1dy <- glc(response ~ y, data=d2, category=d2$category, zlimit=7)
#or using update()
#fit.1dy <- update(fit.2dl, . ~ . -x)

#fit a 2d optimal model
fit.2dlopt <- glc(response ~ x + y, data=d2, category=d2$category, zlimit=7,
  fixed=list(coeffs=TRUE, bias=TRUE))

#calculate AIC and compare
AIC(fit.2dl, fit.1dy, fit.2dlopt)
```

---

glcStruct

*General Linear Classifier structure*

---

**Description**

A named list of model parameters that specify a linear decision bound, containing noise, coeffs, and bias.

**Usage**

```
glcStruct(noise, coeffs, bias)
```

**Arguments**

noise	a positive non-zero numeric.
coeffs	vector. the length of the coeffs should correspond to the number of the model's dimension.
bias	numeric.

**Value**

object of class `glcStruct`, i.e., a named list containing noise, coeffs, and bias. Returned values are normalized, such that each value are divided by the euclidean norm of the coeffs vector, and the sum of coeffs<sup>2</sup> is 1.

**See Also**

[glc](#), [coef.glcStruct](#), [logLik.glcStruct](#), [old2new\\_par](#), [new2old\\_par](#)

**Examples**

```
params <- glcStruct(noise=10, coeffs=c(1, -1), bias=0)
```

gqc

*General Quadratic Classifier***Description**

Fit a general quadratic classifier (a.k.a. quadratic decision-bound model).

**Usage**

```
gqc(formula,
     data,
     category,
     par = list(),
     zlimit = Inf,
     fixed = list(),
     opt = c("nlminb", "optim"),
     lower=-Inf,
     upper=Inf,
     control=list())
```

**Arguments**

formula	A formula of the form $\text{response} \sim x_1 + x_2 + \dots$ where the response specifies the grouping factor (generally a participant's response) and the right hand side specifies the feature values of the classified stimuli.
data	A data frame from which variables specified in formula are taken.
category	(Optional.) A factor specifying the true category membership of the stimuli.
par	object of class <code>gqcStruct</code> or named list containing a set of initial parameters used to fit the data.
zlimit	numeric. The z-scores (or discriminant scores) beyond the specified value will be truncated. Default to <code>Inf</code>
fixed	A named list of logical vectors specifying whether each of <code>pnoise</code> , <code>cnoise</code> , <code>coeffs</code> , and <code>bias</code> parameters should be fixed to the initial value. Default to all <code>FALSE</code> . A fatal error will result if set to all <code>TRUE</code> .
opt	A character string specifying the optimizer to be used: either <code>nlminb</code> (the default) or <code>optim</code> . If "optim", "L-BFGS-B" method is used (see 'Details' of <code>optim</code> )
lower, upper	Bounds on the parameters. Default values of lower and upper are <code>c(.1, .1, rep(-Inf, length(unlist(par))-2))</code> , and <code>c(5000, 5000, rep( Inf, length(unlist(par))-2))</code> , respectively.
control	A list of control parameters passed to the optimizer. See 'Details' of <code>nlminb</code> or <code>optim</code>

## Details

If `par` is not fully specified in the argument, the function attempts to calculate the initial parameter values by internally calling the functions `mcovs` and `qdb`. The response specified in the formula is used as the grouping factor in `mcovs`.

## Value

object of class `gqc`, i.e., a list containing the following components:

<code>terms</code>	the <a href="#">terms</a> object used.
<code>call</code>	the matched call.
<code>model</code>	the design matrix used to fit the model.
<code>category</code>	the category vector as specified in the input.
<code>initpar</code>	the initial parameter used to fit the model.
<code>par</code>	the fitted parameter.
<code>logLik</code>	the log-likelihood at convergence.

## References

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

Ashby, F. G., & Gott, R. E. (1988). *Decision rules in the perception and categorization of multidimensional stimuli*. Journal of Experimental Psychology: Learning, Memory, & Cognition, 14, 33-53.

Ashby, F. G. (1992) *Multidimensional models of perception and cognition*. Lawrence Erlbaum Associates.

## See Also

[glc](#), [qdb](#), [logLik.gqc](#), [logLik.gqcStruct](#), [plot.gqc](#), [plot3d.gqc](#)

## Examples

```
data(subjdemo_2d)
fit.2dq <- gqc(response ~ x + y, data=subjdemo_2d,
              category=subjdemo_2d$category, zlimit=7)
```

---

gqcStruct

*General Quadratic Classifier structure.*


---

**Description**

A named list of model parameters that specify a quadratic decision bound, containing pnoise, cnoise, coeffs, and bias.

**Usage**

```
gqcStruct(pnoise, cnoise, coeffs, bias)
```

**Arguments**

pnoise	a positive non-zero numeric.
cnoise	a positive non-zero numeric.
coeffs	a vector. The length(coeffs) should be equal to $\text{sum}(1:(\text{dim}+1)) - 1$ where dim is the number of the model's dimension
bias	numeric.

**Value**

object of class gqcStruct, i.e., a named list containing pnoise, cnoise, coeffs, and bias.

**See Also**

[gqc](#), [logLik.gqcStruct](#)

**Examples**

```
params <- gqcStruct(pnoise=10, cnoise=100, coeffs=c(1,2,3,4,5), bias=6)
```

---

grg

*General Random Guessing model*


---

**Description**

General Random Guessing model

**Usage**

```
grg(response, fixed = FALSE, k = 2)
```

**Arguments**

response	A vector containing participant's classification responses.
fixed	logical. If TRUE, Fixed Random Guessing model is fitted. If FALSE, General Random Guessing model is fitted. see 'Details'
k	numeric. the penalty per parameter to be used in calculating AIC. Default to 2.

**Details**

The function assumes that there are two categories (e.g, 'A' and 'B') to which each stimulus belongs.

Fixed Random Guessing model assumes that participant responded randomly without response bias; for each stimulus, probability of responding 'A' and 'B' is .5. There are no free parameters in this model (i.e.,  $df = 0$ ).

General Random Guessing model assumes that participants responded randomly but is biased toward one response. The model estimates the response bias ( $df = 1$ ).

**Value**

object of class grg, which is a list object containing:

par	the fixed or estimated response bias
logLik	the log-likelihood of the model
AIC	Akaike's An Information Criterion for the fitted model

**References**

Ashby, F. G., & Crossley, M. J. (2010). *Interactions between declarative and procedural-learning categorization systems*. *Neurobiology of Learning and Memory*, 94, 1-12.

**See Also**

[glc](#), [gqc](#), [extractAIC.grg](#)

**Examples**

```
data(subjdemo_2d)
fit.grand <- grg(subjdemo_2d$response, fixed=FALSE)

fit.frand <- grg(subjdemo_2d$response, fixed=TRUE)
```

---

grtMeans	<i>Obtain means of two multivariate normal populations satisfying certain criteria</i>
----------	--

---

### Description

Obtain means of two multivariate normal populations having the specified covariance structure and centroid, and with which classification based on the optimal decision boundary satisfies the supplied probability of correct classification.

### Usage

```
grtMeans(covs, centroid, optldb, p.correct, initd = 5, stepsize = 1)
```

### Arguments

covs	a matrix or a list of matrices specifying the covariance matrices of the variables. Each matrix should be positive-definite and symmetric.
centroid	a vector specifying the center of the two population means
optldb	object of class <code>glcStruct</code> or a vector of coefficients for the optimal linear decision bound.
p.correct	a numeric value between 0 to 1 that specifies the optimal classification performance in terms of probability of correct classification given the decision boundary <code>optbnd</code> .
initd	numeric. An initial distance between the means of two populations. Default is 5.
stepsize	a positive numeric specifying step size to be taken when searching for the means. Default is 1.

### Value

means	a list of two vectors specifying the means of two populations.
covs	a matrix of (averaged) covariance.
p.correct	the obtained probability of correct classification.

### Author(s)

Author of the original Matlab routine ‘Design2dGRTExp’: Leola Alfonso-Reese

Author of R adaptation: Kazunaga Matsuki

### References

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

**See Also**[ldb.p.correct](#)**Examples**

```
foo <- grtMeans(diag(c(625,625)), centroid=c(200, 200*.6),
  optldb=c(.6,-1,0), p.correct=.85)
```

---

`grtrnorm`*Sample from multiple multivariate normal distributions*

---

**Description**

Generate one or more samples from the two or more specified multivariate normal distributions.

**Usage**

```
grtrnorm(n,
  np = 2,
  means = list(rep(0,np), rep(0,np)),
  covs = diag(rep(1,np)),
  clip.sd = Inf,
  tol = 1e-6,
  empirical = TRUE,
  seed = NULL,
  response.acc = NULL)
```

**Arguments**

<code>n</code>	the number of samples per population required
<code>np</code>	the number of populations to be sampled from
<code>means</code>	a list of vectors specifying the means of the variable for each populations
<code>covs</code>	a matrix or a list of matrices specifying the covariance matrices of the variables. Each matrix should be positive-definite and symmetric.
<code>clip.sd</code>	an integer specifying the cutoff value of standard score. The standard score of a generated sample exceeding this value should be truncated. Default to Inf (no truncation).
<code>tol</code>	tolerance (relative to largest variance) for numerical lack of positive-definiteness in covs.
<code>empirical</code>	logical. If true, means and covs specify the empirical rather than population means and covariance matrices.
<code>seed</code>	an integer internally supplied as seed argument to the function <code>set.seed</code> . If NULL, <code>.Random.seed</code> is used.
<code>response.acc</code>	an optional numeric value between 0 and 1, specifying the classification accuracy of a hypothetical observer. See 'Details'. Default to NULL.

**Details**

This function is essentially a wrapper to the `mvrnorm` function in MASS package.

If the optional response `.acc` argument is supplied, hypothetical random classification responses with specified accuracy will be generated.

**Value**

a data frame containing a column of numeric category labels and column(s) of sampled values for each variable, and optionally, a column of hypothetical responses.

**Author(s)**

Author of the original Matlab routines: Leola Alfonso-Reese

Author of R adaptation: Kazunaga Matsuki

**References**

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

**Examples**

```
m <- list(c(268,157), c(332, 93))
covs <- matrix(c(4538, 4351, 4351, 4538), ncol=2)
II <- grtrnorm(n=80, np=2, means=m, covs=covs)
```

```
m <- list(c(283,98),c(317,98),c(283,152),c(317,152))
covs <- diag(75, ncol=2, nrow=2)
CJ <- grtrnorm(n=c(8,16,16,40), np=4, means=m, covs=covs)
CJ$category <- c(1,1,1,2)[CJ$category]
```

---

 ldb

---

*Linear Decision Bound*


---

**Description**

Find coefficients of the ideal linear decision boundary given the means and covariance of two categories.

**Usage**

```
ldb(means, covs,
    covstruct = c("unstructured", "scaledIdentity", "diagonal", "identity"),
    noise = 10)
```

## Arguments

means	a list of vectors containing means of two distributions.
covs	a matrix or a list of matrix containing the covariance matrix common to the two distributions.
covstruct	character. If "unstructured", the supplied covs is used as-is. If "scaledIdentity", a diagonal covariance matrix with one common variance is used; when the supplied covs has different values on its diagonal, the mean of the diagonal is used. If "diagonal", a diagonal covariance matrix with varying diagonal is used. If "identity", an identity matrix is used. Default to "unstructured"
noise	numeric value. See Details. Default to 10.

## Details

The order of vectors in the list means matters as the sign of coeffs and bias in the output will be reversed.

The argument noise is only for convenience; the supplied value is simply bypassed to the output for the subsequent use, i.e., as object of class `glcStruct`.

## Value

The object of class `glcStruct`

## Author(s)

Author of the original Matlab routine 'lndecisbnd': Leola Alfonso-Reese

Author of R adaptation: Kazunaga Matsuki

## References

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

## See Also

`mcovs`, `qdb`, `glcStruct`, `glc`

## Examples

```
m <- list(c(187, 142), c(213.4, 97.7))
covs <- diag(c(625, 625))
foo <- ldb(means=m, covs=covs)
```

---

ldb.p.correct	<i>Probability of correct classification based on the optimal linear decision bound.</i>
---------------	--

---

### Description

Estimates the probability of correct classification under the condition in which the optimal linear decision boundary is used to categorize the samples from two multivariate normal populations with the specified parameters

### Usage

```
ldb.p.correct(means, covs, noise = 0)
```

### Arguments

means	a list of vectors, each specifying the means of a multivariate normal population.
covs	a matrix or a list of matrices specifying the covariance matrix of the each multivariate normal population. If a list is given and $\text{length}(\text{covs}) > 2$ , an unweighted average of the matrices is used.
noise	an optional numeric value specifying the noise associated with the decision bound. Default to 0.

### Author(s)

Author of the original Matlab routine 'linprobcrr': Leola Alfonso-Reese

Author of R adaptation: Kazunaga Matsuki

### References

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

### Examples

```
foo <- grtMeans(diag(c(625,625)), centroid=c(200, 200*.6),  
  optldb=c(.6,-1,0), p.correct=.85)  
ldb.p.correct(foo$means, foo$covs)
```

---

lines.gqcStruct      *lines Method for class 'gqc'*

---

### Description

Add a quadratic decision boundary line through the current plot.

### Usage

```
## S3 method for class 'gqcStruct'
lines(x,
      xlim = c(0,1), ylim = c(0,1),
      npoints = 100, col = "black",
      ...)
```

### Arguments

x	object of class gqcStruct
xlim	the x limits of the plot. Default to c(0, 1)
ylim	the y limits of the plot. Default to c(0, 1)
npoints	numeric. number of points per dimension used to plot the decision bound. Default is 100.
col	the color to be used for the line
...	further arguments.

### Value

an invisible list of x- and y-coordinates of the line:

x	a vector of x-coordinates of the line
y	a vector of y-coordinates of the line

### See Also

[plot.gqc](#), [{plot3d.gqc}](#)

### Examples

```
data(subjdemo_2d)
fit.2dq <- gqc(response ~ x + y, data=subjdemo_2d,
              category=subjdemo_2d$category, zlimit=7)
plot(fit.2dq, fitdb=FALSE, initdb=FALSE)
lines(fit.2dq$par, xlim=c(0,400), ylim=c(0,400), col="red")
lines(fit.2dq$initpar, xlim=c(0,400), ylim=c(0,400), col="blue")
```

---

`logLik.glc`*Log-Likelihood of a 'glc' or 'gcjc' Object*

---

**Description**

Extract the log-likelihood of the fitted general linear or conjunctive classifier model.

**Usage**

```
## S3 method for class 'glc'  
logLik(object, ...)  
  
## S3 method for class 'gcjc'  
logLik(object, ...)
```

**Arguments**

<code>object</code>	object of class <code>glc</code> or <code>gcjc</code>
<code>...</code>	further arguments (currently unused)

**Value**

The log-likelihood for the general linear or conjunctive classifier represented by the estimated parameters in `object`

**Note**

This function is intended for indirect internal use by functions such as AIC. To obtain the log-likelihood of the fitted model applied to new dataset, use [logLik.glcStruct](#) or [logLik.gcjcStruct](#)

**See Also**

[glc](#), [logLik.glcStruct](#), [gcjc](#), [logLik.gcjcStruct](#)

**Examples**

```
data(subjdemo_2d)  
fit <- glc(response ~ x + y, data=subjdemo_2d,  
           category=subjdemo_2d$category, zlimit=7)  
logLik(fit)
```

---

logLik.glcStruct	<i>Log-Likelihood of a 'glcStruct' or 'gcjcStruct' Object</i>
------------------	---

---

**Description**

Calculate the log-likelihood of the general linear or conjunctive classifier model applied to a data set.

**Usage**

```
## S3 method for class 'glcStruct'
logLik(object, response, x, zlimit = Inf, ...)
```

```
## S3 method for class 'gcjcStruct'
logLik(object, response, x, zlimit = Inf, ...)
```

**Arguments**

object	object of class glcStruct or gcjcStruct containing the parameter values at which the log-likelihood is to be evaluated.
response	a vector of classification responses used to calculate the log-likelihood of the model.
x	a matrix or dataframe containing values for each stimulus dimensions.
zlimit	integer. Used to truncate the z-scores whose absolute values are greater than zlimit when calculating the log-likelihood. Default to Inf
...	further arguments (currently unused)

**Value**

The log-likelihood for the general linear or conjunctive classifier described by object fitted against the dataset given by response and x.

**Note**

The value of attributes, attr(, "df") (degrees of freedom) is calculated based on the assumption that all the parameters in object are free to vary.

**See Also**

[gqc](#), [gqcStruct](#), [logLik.glc](#), [logLik.gcjc](#)

**Examples**

```
m <- list(c(187, 142), c(213, 98))
covs <- diag(625, ncol=2, nrow=2)
db <- ldb(means=m, covs=covs, noise=10)
data(subjdemo_2d)
logLik(db, subjdemo_2d$response, x=subjdemo_2d[2:3], zlimit=7)
```

---

logLik.gqc	<i>Log-Likelihood of a 'gqc' Object</i>
------------	---

---

**Description**

Extract the log-likelihood of the fitted general quadratic classifier model.

**Usage**

```
## S3 method for class 'gqc'
logLik(object, ...)
```

**Arguments**

object	object of class gqc
...	further arguments (currently unused)

**Value**

The log-likelihood for the general quadratic classifier represented by the estimated parameters in object

**Note**

This function is intended for indirect internal use by functions such as AIC. To obtain the log-likelihood of the fitted model applied to new dataset, use [logLik.gqcStruct](#)

**See Also**

[gqc](#), [logLik.gqcStruct](#)

---

logLik.gqcStruct	<i>Log-Likelihood of a 'gqcStruct' Object</i>
------------------	---

---

**Description**

Calculate the log-likelihood of the general quadratic classifier model applied to a data set.

**Usage**

```
## S3 method for class 'gqcStruct'
logLik(object, response, x, zlimit = Inf, ...)
```

**Arguments**

object	object of class <code>gqcStruct</code>
response	a vector of classification responses used to calculate the log-likelihood of the <code>gqc</code> model.
x	a matrix or dataframe containing values for each stimulus dimensions.
zlimit	integer. Used to truncate the z-scores whose absolute values are greater than <code>zlimit</code> when calculating the log-likelihood. Default to <code>Inf</code>
...	further arguments (currently unused)

**Value**

The log-likelihood for the general quadratic classifier described by object fitted against the dataset given by response and x.

**Note**

The value of attributes, `attr(, "df")` (degrees of freedom) is calculated based on the assumption that all the parameters in object are free to vary.

**See Also**

[gqc](#), [gqcStruct](#), [logLik.gqc](#)

**Examples**

```
m <- list(c(187, 142), c(213, 98))
covs <- list(diag(625, ncol=2, nrow=2), diag(600, ncol=2, nrow=2))
db <- qdb(means=m, covs=covs)
data(subjdemo_2d)
logLik(db, subjdemo_2d$response, x=subjdemo_2d[2:3], zlimit=7)
```

---

mcovs

---

*Calculate sample means and covariance(s) of multivariate data*


---

**Description**

Calculate sample means and covariance(s) of multivariate data

**Usage**

```
## Default S3 method:
mcovs(x, grouping, pooled=TRUE, ...)

## S3 method for class 'formula'
mcovs(formula, data, pooled=TRUE, ...)
```

**Arguments**

formula	A formula in the form of grouping $\sim x_1 + x_2 + \dots$ , where the right hand side specifies the sample variables.
data	Data frame from which variables specified in formula are taken.
x	data frame or Matrix containing sample values.
grouping	a factor specifying the population to which the samples in x belong.
pooled	logical. If TRUE, pooled variance-covariance matrix is calculated. If FALSE, a list of variance-covariance matrices for each groups are calculated. Default to TRUE.
...	further arguments

**Value**

A list containing:

N	total number of samples.
counts	number of samples per groups.
lev	levels of the grouping factor
means	a named list of vectors specifying the means for each group. Named according to lev.
covs	a named list of variance-covariance matrix(es). Named as pooled if pooled=TRUE, otherwise according to lev.

---

new2old\_par

*Convert 'new' to 'old' glcStruct format*

---

**Description**

Converts the glcStruct in 'new' format to 'old' format whereby a vector of angle is converted to coeffs.

**Usage**

new2old\_par(x)

angle2cart(angle)

**Arguments**

x	object of class glcStruct.
angle	vector.

**Value**

For new2old\_par, object of class glcStruct.

For angle2cart, vector.

**Author(s)**

Author of the original Matlab routines: Leola Alfonso-Reese

Author of R adaptation: Kazunaga Matsuki

**References**

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

**See Also**

[old2new\\_par](#) [cart2angle](#) [glcStruct](#)

**Examples**

```
m <- list(c(187, 142), c(213.4, 97.7))
covs <- diag(c(625, 625))
foo <- ldb(means=m, covs=covs)
bar <- old2new_par(foo)
new2old_par(bar)

angle2cart(bar$angle)
```

---

old2new\_par

*Convert 'old' to 'new' glcStruct format*

---

**Description**

Converts glcStruct in the 'old' to 'new' format for more efficient optimization where coeffs vectors are converted to a vector of angle with length of (coeffs) - 1

**Usage**

```
old2new_par(x)
```

```
cart2angle(cart)
```

**Arguments**

x                    object of class glcStruct.

cart                 vector.

**Value**

For old2new\_par, object of class glcStruct.  
 For cart2angle, vector.

**Author(s)**

Author of the original Matlab routines: Leola Alfonso-Reese  
 Author of R adaptation: Kazunaga Matsuki

**References**

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

**Examples**

```
m <- list(c(187, 142), c(213.4, 97.7))
covs <- diag(c(625, 625))
foo <- ldb(means=m, covs=covs)
old2new_par(foo)

cart2angle(foo$coeffs)
```

---

 plot.gcjc

---

*Plot Method for Class 'gcjc'*


---

**Description**

Plot the fitted data set and decision boundary.

**Usage**

```
## S3 method for class 'gcjc'
plot(x, fitdb = TRUE, initdb = FALSE, xlim = NULL, ylim = NULL, bg, pch, ...)
```

**Arguments**

x	object of class gcjc
fitdb	logical. If TRUE, the fitted decision bound is plotted. Default to TRUE
initdb	logical. If TRUE, the decision bound specified by the initial parameters is plotted. Default to FALSE
xlim	the x limits of the plot
ylim	the y limits of the plot
bg	the background color to be used for points. Default is c("white", "gray")[response] where response represents the response vector of the model
pch	the symbols to be used as points. Default is c(21, 24)[x\$category]
...	further arguments.

**Details**

This function produces a scatter plot of data matrix in the  $x$  and (optionally) decision boundary specified within (i.e.,  $x\$par$  and/or  $x\$initpar$ ).

**Examples**

```
m <- list(c(100,200),c(100,100),c(200,100),c(200,200))
covs <- diag(30^2, ncol=2, nrow=2)
set.seed(1)
CJ <- grtrnorm(n=c(50,20,10,20), np=4, means=m, covs=covs)
CJ$category <- c(1,2,2,2)[CJ$category]
#create random responses with 80% accuracy
CJ$response <- CJ$category
set.seed(1)
incorrect <- sample(1:100, size=20)
CJ$response[incorrect] <- abs(CJ$response[incorrect] - 3)

#now fit the model
m.cj <- gcjc(response ~ x1 + x2, data=CJ, config=2, category=CJ$category, zlimit=7)

plot(m.cj)
```

---

plot.glc

*Plot Method for Class 'glc'*


---

**Description**

Plot the fitted data set and linear decision boundary.

**Usage**

```
## S3 method for class 'glc'
plot(x, fitdb = TRUE, initdb = FALSE, xlim = NULL, ylim = NULL, bg, pch, ...)
```

**Arguments**

<code>x</code>	object of class <code>glc</code>
<code>fitdb</code>	logical. If TRUE, the fitted decision bound is plotted. Default to TRUE
<code>initdb</code>	logical. If TRUE, the decision bound specified by the initial parameters is plotted. Default to FALSE
<code>xlim</code>	the x limits of the plot
<code>ylim</code>	the y limits of the plot
<code>bg</code>	the background color to be used for points. Default is <code>c("white", "gray")[response]</code> where response represents the response vector of the model
<code>pch</code>	the symbols to be used as points. Default is <code>c(21, 24)[x\$category]</code>
<code>...</code>	further arguments.

**Details**

This function produces a scatter plot of data matrix in the  $x$  and (optionally) decision boundary specified within (i.e.,  $x\$par$  and/or  $x\$initpar$ ).

The look of the plot differs depending on the dimension of the model. If the dimension is 1, the model matrix is plotted on the  $y$ -axis, and category vector (as in  $x$category$ ) is plotted on the  $x$  axis. If the dimension is 2, scatter plot of the model matrix is plotted. If the dimension is 3, `plot3d.glc` is called to create a 3D scatter plot. If the dimension is greater than 3, an error message will be returned.

**See Also**

[plot3d.glc](#)

**Examples**

```
data(subjdemo_2d)
fit.2dl <- glc(response ~ x + y, data=subjdemo_2d,
              category=subjdemo_2d$category, zlimit=7)
plot(fit.2dl)

#if one wants to plot decision bounds in
# colors different from the defaults
plot(fit.2dl, fitdb=FALSE)
abline(coef=coef(fit.2dl$par), col="orange")
abline(coef=coef(fit.2dl$initpar), col="purple")
```

---

plot.gqc

*plot Method for Class 'gqc'*

---

**Description**

Plot the fitted data set and quadratic decision boundary.

**Usage**

```
## S3 method for class 'gqc'
plot(x, fitdb = TRUE, initdb = FALSE,
     xlim = NULL, ylim = NULL, bg, pch, npoints = 100, ...)
```

**Arguments**

<code>x</code>	object of class <code>gqc</code>
<code>fitdb</code>	logical. If TRUE, the fitted decision bound will be plotted. Default to TRUE.
<code>initdb</code>	logical. If TRUE, the decision bound specified by the initial parameters will be plotted. Default to FALSE.

xlim	the x limits of the plot. If NULL, limits are calculated from the model matrix. Default to NULL.
ylim	the y limits of the plot. If NULL, limits are calculated from the model matrix. Default to NULL.
bg	the background color to be used for points in 2D scatter plot. Default is c("white", "gray")[response] where response represents the response vector of the model
pch	the symbols to be used as points in 2D scatter plot. Default is c(21, 24)[x\$category]
npoints	number of points per dimension used to plot the decision bound. Default is 100.
...	further arguments.

### Details

This function produces a scatter plot of data matrix in the x and (optionally) decision boundary (i.e., x\$par and/or x\$initpar).

The look of the plot differs depending on the dimension of the model. If the dimension is 2, scatter plot of the model matrix is plotted. If the dimension is 3, plot3d.glc is called to create a 3D scatter plot. In all other cases, an error message will be returned.

### See Also

[lines.gqcStruct](#), {[plot3d.gqc](#)}

### Examples

```
data(subjdemo_2d)
fit.2dq <- gqc(response ~ x + y, data=subjdemo_2d,
  category=subjdemo_2d$category, zlimit=7)
plot(fit.2dq)
```

---

plot3d.glc

*plot3d Method for Class 'glc'*

---

### Description

plot the fitted 3D data set and linear decision boundary.

### Usage

```
## S3 method for class 'glc'
plot3d(x, fitdb = TRUE, initdb = FALSE,
  lims = NULL, alpha = .5, ...)
```

## Arguments

x	object of class <code>glc</code>
fitdb	logical. If TRUE, the fitted decision bound will be plotted. Default to TRUE
initdb	logical. If TRUE, the decision bound specified by the initial parameters will be plotted. Default to FALSE
lims	column matrix of the x, y, and z limits of the plot
alpha	alpha value for the decision bound surface(s) ranging from 0 (fully transparent) to 1 (opaque). Default is .5.
...	further arguments.

## Details

This function produces a 3D scatter plot of data matrix in the x and (optionally) decision boundary specified within (i.e., `x$par` and/or `x$initpar`), using `points3d` and `quads3d` in the `rgl` package respectively.

## References

Daniel Adler, Oleg Nenadic and Walter Zucchini (2003) RGL: A R-library for 3D visualization with OpenGL

## See Also

[plot.glc](#), [plot3d.gqc](#)

## Examples

```
## Not run:
data(subjdemo_3d)
fit.3d1 <- glc(response ~ x + y + z, data=subjdemo_3d,
  category=subjdemo_3d$category, zlimit=7)
plot3d(fit.3d1)

## End(Not run)
```

---

plot3d.gqc

*plot3d Method for Class 'gqc'*

---

## Description

plot the fitted 3D data set and quadratic decision boundaries.

**Usage**

```
## S3 method for class 'gqc'
plot3d(x, fitdb = TRUE, initdb = FALSE,
       lims = NULL, npoints = 100, alpha = .5,
       fill = TRUE, smooth = FALSE, ...)
```

**Arguments**

x	object of class gqc
fitdb	logical. If TRUE, the fitted decision bound will be plotted. Default to TRUE
initdb	logical. If TRUE, the decision bound specified by the initial parameters will be plotted. Default to FALSE
lims	column matrix of the x, y, and z limits of the plot. If NULL, limits are calculated from the model matrix of x. Default to NULL.
npoints	number of points per dimension (i.e., x, y, and z) used to plot the decision bound surface. Default is 100.
alpha	alpha value for the decision bound surface(s) ranging from 0 (fully transparent) to 1 (opaque). Default is .5
fill	logical. If TRUE, decision bounds (if fitdb or initdb is set to TRUE) should be drawn with filled surfaces. If FALSE, a wire frame should be used. Default to TRUE.
smooth	logical. If TRUE, smooth shading should be used. Default to FALSE.
...	further arguments.

**Details**

This function produces a 3D scatter plot of data matrix of x and (optionally) quadratic decision boundaries specified within (i.e., x\$par and/or x\$initpar), using points3d function in the rgl package and contour3d function in the misc3d package respectively.

**References**

Daniel Adler, Oleg Nenadic and Walter Zucchini (2003) RGL: A R-library for 3D visualization with OpenGL

**See Also**

[plot.gqc](#), [{plot3d.gqc}](#)

**Examples**

```
## Not run:
data(subjdemo_3d)
fit.3dq <- gqc(response ~ x + y + z, data=subjdemo_3d,
              category=subjdemo_3d$category, zlimit=7)
plot3d(fit.3dq)
```

```
## End(Not run)
```

---

predict.glc

*predict method for General Linear Classifier*

---

### Description

Predicted classification based on 'glc' model object.

### Usage

```
## S3 method for class 'glc'  
predict(object, newdata, seed = NULL, ...)
```

### Arguments

object	object of class glc.
newdata	a vector or a matrix containing new samples with which the classification prediction is to be made.
seed	numeric. The 'seed' used for the random number generator.
...	further arguments (currently unused).

### Details

The function predict (or 'simulate') classification response of an observer whose noise and linear decision bounds are specified in object.

The predicted category labels are matched with those used for the fit in object.

If newdata is missing, the predictions are made on the data used for the fit.

### Value

a vector of labels of categories to which each sample in newdata is predicted to belong, according to the model in object.

### Author(s)

Author of the original Matlab routines: Leola Alfonso-Reese

Author of R adaptation: Kazunaga Matsuki

### References

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

## Examples

```
data(subjdemo_2d)
fit.2dl <- glc(response ~ x + y, data=subjdemo_2d,
              category=subjdemo_2d$category, zlimit=7)

m <- list(c(187, 142), c(213.4, 97.7))
covs <- diag(c(900, 900))
newd <- grtrnorm(n=20, np=2, means=m, covs=covs, seed=1234)
predict(fit.2dl, newd[,2:3], seed=1234)
```

---

qdb

*Quadratic Decision Bound*

---

## Description

Find coefficients of the ideal quadratic decision boundary given the means and covariance of two categories.

## Usage

```
qdb(means, covs, pnoise = 10, cnoise = 100, sphere = FALSE)
```

## Arguments

means	a list of vectors containing means of the two distributions.
covs	a list containing the covariance matrices of the two distributions.
pnoise, cnoise	numeric. Defaults set to 10, and 100, respectively. see 'Details'
sphere	logical. If TRUE, the returned decision bound forms a circle or sphere.

## Details

The order of vectors in the list means and covs matters as the sign of coeffs and bias object in the output will be reversed.

The argument pnoise and cnoise is only for convenience; the supplied value is simply bypassed to the output for the subsequent use, i.e., as object of class gqcStruct.

## Value

object of class gqcStruct

## Author(s)

Author of the original Matlab routine 'quaddecisbnd': Leola Alfonso-Reese

Author of R adaptation: Kazunaga Matsuki

**References**

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

**See Also**

[mcovs](#), [qdb](#), [gqcStruct](#), [gqc](#)

**Examples**

```
m <- list(c(187, 142), c(213.4, 97.7))
covs <- list(diag(c(625, 625)), diag(c(625, 625)))
foo <- qdb(means=m, covs=covs)
```

---

qdb.p.correct	<i>the proportion correct of the quadratic decision boundary.</i>
---------------	---

---

**Description**

Calculate the proportion correct obtained by categorizing samples from one multivariate normal population using the quadratic decision boundary.

**Usage**

```
qdb.p.correct(x, qdb, refpts = colMeans(x))
```

**Arguments**

x	a vector or matrix containing the values of samples from one multivariate normal population.
qdb	object of class <code>gqcStruct</code> or a vector containing the values for <code>coeffs</code> and <code>bias</code> of an quadratic decision bound.
refpts	a numeric vector used as a reference point to determine the correct side of the qdb for classifying x. The <code>length(refpts)</code> should be equal to <code>ncol(x)</code> . Default to <code>colMeans(x)</code> .

**Details**

The function assumes that all the points specified in x belong to just one category.

**Author(s)**

Author of the original Matlab routine 'quadbndpercrr': Leola Alfonso-Reese

Author of R adaptation: Kazunaga Matsuki

## References

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

## Examples

```
data(subjdemo_2d)
tmp <- split(subjdemo_2d, subjdemo_2d$category)
mc <- mcovs(category ~ x + y, data=subjdemo_2d, pooled=FALSE)
db <- qdb(mc$means, mc$covs)
qdb.p.correct(tmp[[1]][,2:3], db)
```

---

scale

*Scale method for the class 'glc' and 'gqc'*

---

## Description

Return the discriminant scores obtained by applying the general linear classifier to the fitted data.

## Usage

```
## S3 method for class 'glc'
scale(x, initdb = FALSE, zlimit = Inf, ...)
## S3 method for class 'gqc'
scale(x, initdb = FALSE, zlimit = Inf, ...)
```

## Arguments

x	object of class glc or gqc
initdb	optional logical. If TRUE, the returned vector represents the z-scores with respect to the initial parameters, rather than the fitted parameters. Defaults to FALSE.
zlimit	optional numeric. Used to truncate the scores beyond the specified value. Default to Inf
...	further arguments (currently unused)

## Note

The generic function `scale` is redefined to accept arguments other than `x`, `center`, and `scale`.

## Examples

```
data(subjdemo_2d)
fit.2d1 <- glc(response ~ x + y, data=subjdemo_2d,
  category=subjdemo_2d$category, zlimit=7)
scale(fit.2d1)

fit.2dq <- gqc(response ~ x + y, data=subjdemo_2d,
```

```
category=subjdemo_2d$category, zlimit=7)
scale(fit.2dq)

## Not run:
#plots using the discriminant scores
require(Hmisc)
options(digits=3)
fit.2dl <- glc(response ~ x + y, data=subjdemo_2d,
  category=subjdemo_2d$category, zlimit=7)
# z-scores based on the initial decision bound
# split by the true category membership
zinit <- split(scale(fit.2dl, initdb=TRUE), subjdemo_2d$category)
histbackback(zinit)

# z-scores based on the fitted decision bound
# split by the participants' response
zfit1 <- split(scale(fit.2dl, initdb=FALSE), subjdemo_2d$category)
histbackback(zfit1)

# z-scores based on the fitted decision bound
# split by the true category membership
zfit2 <- split(scale(fit.2dl, initdb=FALSE), subjdemo_2d$response)
histbackback(zfit2)

## End(Not run)
```

---

subjdemo\_1d

*Sample dataset of a categorization experiment with 1D stimuli.*

---

## Description

A sample one dimensional stimulus set and response data of a hypothetical participant in a two-category classification experiment involving 500 trials.

## Usage

```
subjdemo_1d
```

## Format

This data frame contains 500 rows and the following columns:

category label of the category to which each stimulus belongs.

x value on the dimension 'x'

response classification response of a participant.

**Source**

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

**References**

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

---

subjdemo\_2d

*Sample dataset of a categorization experiment with 2D stimuli.*

---

**Description**

A sample two dimensional stimulus set and response data of a hypothetical participant in a two-category classification experiment involving 500 trials.

**Usage**

subjdemo\_2d

**Format**

This data frame contains 500 rows and the following columns:

category label of the category to which each stimulus belongs.

x value on the dimension 'x'

y value on the dimension 'y'

response classification response of a participant.

**Source**

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

**References**

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

---

`subjdemo_3d`*Sample dataset of a categorization experiment with 3D stimuli.*

---

**Description**

A sample one dimensional stimulus set and response data of a hypothetical participant in a two-category classification experiment involving 500 trials.

**Usage**`subjdemo_3d`**Format**

This data frame contains 500 rows and the following columns:

`category` label of the category to which each stimulus belongs.

`x` value on the dimension 'x'

`y` value on the dimension 'y'

`z` value on the dimension 'z'

`response` classification response of a participant.

**Source**

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

**References**

Alfonso-Reese, L. A. (2006) *General recognition theory of categorization: A MATLAB toolbox*. Behavior Research Methods, 38, 579-583.

---

`subjdemo_cj`*Sample dataset of a categorization experiment with 2D conjunctive stimuli.*

---

**Description**

A sample two dimensional stimulus set and response data of a hypothetical participant in a two-category classification experiment involving 100 trials.

**Usage**`subjdemo_cj`

**Format**

This data frame contains 100 rows and the following columns:

category label of the category to which each stimulus belongs.

x1 value on the dimension 'x1'

x2 value on the dimension 'x2'

response classification response of a participant.

**Examples**

```
##### the data was generated using following codes:
m <- list(c(100,200),c(100,100),c(200,100),c(200,200))
covs <- diag(30^2, ncol=2, nrow=2)
set.seed(1)
subjdemo_cj <- grtrnorm(n=c(50,20,10,20), np=4, means=m, covs=covs)
subjdemo_cj$category <- c(1,2,2,2)[subjdemo_cj$category]
##### create random responses with 80% accuracy
subjdemo_cj$response <- subjdemo_cj$category
set.seed(1)
incorrect <- sample(1:100, size=20)
subjdemo_cj$response[incorrect] <- abs(subjdemo_cj$response[incorrect] - 3)

##### plotting the dataset
with(subjdemo_cj, plot(x2 ~ x1, bg=category, pch=response))
abline(h=150, lty=2)
abline(v=150, lty=2)
```

---

unscale

*Un-scale or re-center the scaled or centered Matrix-like object*

---

**Description**

This function revert a Matrix-like object that is scaled or centered via `scale.default` to data with the original scale/center.

**Usage**

```
unscale(x)
```

**Arguments**

x numeric matrix(like object).

**Value**

a matrix that are re-centered or un-scaled, based on the value of attributes "scaled:center" and "scaled:scale" of x. If neither of those attributes is specified, x is returned.

**See Also**[scale](#)**Examples**

```
require(stats)
x <- matrix(1:10, ncol=2)
unscale(z <- scale(x))

#maybe useful for truncating
trunc <- 1
z[abs(z) > trunc] <- sign(z[abs(z) > trunc])*trunc
unscale(z)
```

# Index

- \* **aplot**
    - lines.gqcStruct, 23
  - \* **array**
    - scale, 39
    - unscale, 43
  - \* **datagen**
    - grtMeans, 18
    - grtrnorm, 19
  - \* **datasets**
    - subjdemo\_1d, 40
    - subjdemo\_2d, 41
    - subjdemo\_3d, 42
    - subjdemo\_cj, 42
  - \* **misc**
    - dprime, 4
    - gaborPatch, 7
    - new2old\_par, 28
    - old2new\_par, 29
  - \* **models**
    - coef.glc, 3
    - extractAIC, 5
    - gcjcStruct, 10
    - glcStruct, 13
    - gqcStruct, 16
    - ldb, 20
    - logLik.glc, 24
    - logLik.glcStruct, 25
    - logLik.gqc, 26
    - logLik.gqcStruct, 26
    - predict.glc, 36
    - qdb, 37
  - \* **multivariate**
    - gcjc, 9
    - glc, 11
    - gqc, 14
    - grg, 16
    - ldb.p.correct, 22
    - mcovs, 27
    - plot.gcjc, 30
    - plot.glc, 31
    - plot.gqc, 32
    - plot3d.glc, 33
    - plot3d.gqc, 34
    - qdb.p.correct, 38
  - \* **package**
    - grt-package, 2
- angle2cart (new2old\_par), 28
- cart2angle, 29
- cart2angle (old2new\_par), 29
- coef.gcjc, 10
- coef.gcjc (coef.glc), 3
- coef.glc, 3, 13
- coef.glcStruct, 11, 13
- coef.glcStruct (coef.glc), 3
- dprime, 4
- dprimef (dprime), 4
- extractAIC, 5
- extractAIC.grg, 17
- gaborPatch, 7
- gcjc, 2, 9, 11, 24
- gcjcStruct, 10
- glc, 2, 10, 11, 13, 15, 17, 21, 24
- glcStruct, 13, 21, 29
- gqc, 2, 13, 14, 16, 17, 25–27, 38
- gqcStruct, 16, 25, 27, 38
- grg, 2, 16
- grt (grt-package), 2
- grt-package, 2
- grtMeans, 2, 18
- grtrnorm, 2, 19
- image, 7
- ldb, 12, 13, 20
- ldb.p.correct, 19, 22

lines.gqcStruct, [23](#), [33](#)  
logLik.gcjc, [10](#), [25](#)  
logLik.gcjc(logLik.glc), [24](#)  
logLik.gcjcStruct, [24](#)  
logLik.gcjcStruct(logLik.glcStruct), [25](#)  
logLik.glc, [13](#), [24](#), [25](#)  
logLik.glcStruct, [11](#), [13](#), [24](#), [25](#)  
logLik.gqc, [15](#), [26](#), [27](#)  
logLik.gqcStruct, [15](#), [16](#), [26](#), [26](#)

mcovs, [21](#), [27](#), [38](#)

new2old\_par, [13](#), [28](#)  
nlminb, [9](#), [12](#), [14](#)

old2new\_par, [13](#), [29](#), [29](#)  
optim, [9](#), [12](#), [14](#)

plot.gcjc, [2](#), [10](#), [30](#)  
plot.glc, [2](#), [13](#), [31](#), [34](#)  
plot.gqc, [2](#), [15](#), [23](#), [32](#), [35](#)  
plot3d.glc, [2](#), [13](#), [32](#), [33](#)  
plot3d.gqc, [2](#), [15](#), [23](#), [33](#), [34](#), [34](#), [35](#)  
predict.glc, [13](#), [36](#)  
print.gcjc(gcjc), [9](#)  
print.glc(glc), [11](#)  
print.gqc(gqc), [14](#)

qdb, [15](#), [21](#), [37](#), [38](#)  
qdb.p.correct, [38](#)

scale, [39](#), [44](#)  
scale.glc, [13](#)  
set.seed, [19](#)  
subjdemo\_1d, [40](#)  
subjdemo\_2d, [41](#)  
subjdemo\_3d, [42](#)  
subjdemo\_cj, [42](#)

terms, [10](#), [12](#), [15](#)

unscale, [43](#)