Package ‘ellipse’

July 20, 2023

Version 0.5.0
Title Functions for Drawing Ellipses and Ellipse-Like Confidence Regions
Author Duncan Murdoch and E. D. Chow (porting to R by Jesus M. Frias Celayeta)
Maintainer Duncan Murdoch <murdoch.duncan@gmail.com>
Depends R (>= 2.0.0),graphics,stats
Suggests MASS
LazyLoad yes
License GPL (>= 2)
URL https://github.com/dmurdoch/ellipse,
       https://dmurdoch.github.io/ellipse/
BugReports https://github.com/dmurdoch/ellipse/issues
NeedsCompilation no
Repository CRAN
Date/Publication 2023-07-20 19:10:02 UTC

R topics documented:

  ellipse-package ........................................  2
  ellipse ..................................................  3
  ellipse.arima0 .........................................  4
  ellipse.glm .............................................  5
  ellipse.lm ..............................................  6
  ellipse.nls ............................................  7
  ellipse.profile .......................................  8
Description

This package contains various routines for drawing ellipses and ellipse-like confidence regions, implementing the plots described in Murdoch and Chow (1996).

There are also routines implementing the profile plots described in Bates and Watts (1988).

Details

There are three groups of routines in the ellipse package. The first consists of those involved with plotcorr, which implements the plots described in Murdoch and Chow (1996). These display correlations using ellipses, whose shape is that of the contours of a bivariate normal distribution with matching correlation.

The second group implements a version of the profile plots described in Bates and Watts (1988); see ellipse.profile and pairs.profile.

The last group provide the basis for the others, drawing ellipses based on various S objects, including scalar correlations, covariance matrices arima, lm, and nls fits: see ellipse.

Author(s)

Duncan Murdoch and E. D. Chow (porting to R by Jesus M. Frias Celayeta.)

Maintainer: Duncan Murdoch <murdoch.duncan@gmail.com>

References


**Description**

A generic function returning an ellipse or other outline of a confidence region for two parameters.

**Usage**

```r
ellipse(x, 
## Default S3 method:
ellipse(x, scale = c(1, 1), centre = c(0, 0), level = 0.95, 
t = sqrt(qchisq(level, 2)), which = c(1, 2), npoints = 100, center = centre, 
...)```

**Arguments**

- `x` An object. In the default method the parameter `x` should be a correlation between -1 and 1 or a square positive definite matrix at least 2x2 in size. It will be treated as the correlation or covariance of a multivariate normal distribution.
- `...` Descendant methods may require additional parameters.
- `scale` If `x` is a correlation matrix, then the standard deviations of each parameter can be given in the scale parameter. This defaults to `c(1, 1)`, so no rescaling will be done.
- `centre` The centre of the ellipse will be at this position.
- `level` The confidence level of a pairwise confidence region. The default is 0.95, for a 95% region. This is used to control the size of the ellipse being plotted. A vector of levels may be used.
- `t` The size of the ellipse may also be controlled by specifying the value of a t-statistic on its boundary. This defaults to the appropriate value for the confidence region.
- `which` This parameter selects which pair of variables from the matrix will be plotted. The default is the first 2.
- `npoints` The number of points used in the ellipse. Default is 100.
- `center` An alternative to `centre` to accommodate US spelling.

**Details**

The default method uses the \((\cos(\theta + d/2), \cos(\theta - d/2))\) parametrization of an ellipse, where \(\cos(d)\) is the correlation of the parameters.

**Value**

An `npoints` x 2 matrix is returned with columns named according to the row names of the matrix `x` (default `‘x’` and `‘y’`), suitable for plotting.
 ellipse.arima0

References

See Also
 ellipse.lm, ellipse.nls, ellipse.profile, ellipse.profile.nls, ellipse.arima0, plotcorr

Examples
# Plot an ellipse corresponding to a 95% probability region for a
# bivariate normal distribution with mean 0, unit variances and
# correlation 0.8.
plot(ellipse(0.8), type = 'l')

ellipse.arima0 Outline an approximate pairwise confidence region

Description
This function produces the ellipsoidal outline of an approximate pairwise confidence region for an ARIMA model fit.

Usage
## S3 method for class 'arima0'
ellipsis(x, which = c(1, 2), level = 0.95, t = sqrt(qchisq(level, 2)), ...)

Arguments
 x The first argument should be an arima0 object, usually resulting from a call to arima0().
 which Which selects the pair of parameters to be plotted. The default is the first two.
 level The confidence level of the region. Default 95%.
 t The t statistic on the boundary of the ellipse.
 ... Other ellipse.default parameters may also be used.

Details
The summary function is used to obtain the approximate covariance matrix of the fitted parameters.

Value
A matrix with columns x and y to outline the confidence region.
ellipse.glm

See Also

ellipse

Examples

data(USAccDeaths)
fit <- arima0(USAccDeaths, order = c(0, 1, 1), seasonal = list(order = c(0, 1, 1)))
# Plot the approximate 95% confidence region for the first two parameters
# of the model
plot(ellipse(fit), type = 'l')
points(fit$coef[1], fit$coef[2])

ellipse.glm

Outline an approximate pairwise confidence region

Description

This function produces the ellipsoidal outline of an approximate pairwise confidence region for a
generalized linear model fit.

Usage

## S3 method for class 'glm'
ellipse(x, which = c(1, 2), level = 0.95, t, npoints = 100,
dispersion, ...)

Arguments

x
which
level
t
npoints
dispersion
...

The first argument should be a glm object, usually resulting from a call to glm().
Which selects the pair of parameters to be plotted. The default is the first two.
The confidence level of the region. Default 95%.
The t statistic on the boundary of the ellipse. For Binomial or Poisson families,
sqrt(qchisq(level, 2)) is used; for other distributions, sqrt(2*qf(level,
df)) where df is the residual degrees of freedom.
How many points to return in the ellipse.
The value of dispersion to use. If specified, it is treated as fixed, and the chi-
square limits for t are used. If missing, it is taken from summary(x).
Other ellipse.default parameters may also be used.

Details

The summary function is used to obtain the approximate covariance matrix of the fitted parameters,
the dispersion estimate, and the degrees of freedom.

Value

A matrix with columns named according to which to outline the confidence region.
See Also

ellipse.default

Examples

```r
## Dobson (1990) Page 93: Randomized Controlled Trial :

counts <- c(18,17,15,20,10,20,25,13,12)
outcome <- gl(3,1,9)
treatment <- gl(3,3)
glm.D93 <- glm(counts ~ outcome + treatment, family=poisson())

# Plot an approximate 95 % confidence region for the two Outcome parameters

plot(ellipse(glm.D93, which = c(2,3)), type = 'l')
points(glm.D93$coefficients[2], glm.D93$coefficients[3])
```

Description

This function produces the ellipsoidal outline of a pairwise confidence region for a linear model fit.

Usage

```r
## S3 method for class 'lm'

ellipse(x, which = c(1, 2), level = 0.95,
        t = sqrt(2 * qf(level, 2, x$df.residual)), ...)
```

Arguments

- `x` The first argument should be an `lm` object, usually resulting from a call to `lm()`.
- `which` Which selects the pair of parameters to be plotted. The default is the first two.
- `level` The confidence level of the region. Default 95%.
- `t` The t statistic on the boundary of the ellipse.
- `...` Other `ellipse.default` parameters may also be used.

Details

The summary function is used to obtain the covariance matrix of the fitted parameters.

Value

A matrix with columns x and y to outline the confidence region.
### ellipse.nls

Outline an approximate pairwise confidence region

### Description

This function produces the ellipsoidal outline of an approximate pairwise confidence region for a nonlinear model fit.

### Usage

```r
## S3 method for class 'nls'
ellipse(x, which = c(1, 2), level = 0.95, 
       t = sqrt(2 * qf(level, 2, s$df[2])), ...)
```

### Arguments

- **x**: The first argument should be an `nls` object, usually resulting from a call to `nls()`.
- **which**: Which selects the pair of parameters to be plotted. The default is the first two.
- **level**: The confidence level of the region. Default 95%.
- **t**: The t statistic on the boundary of the ellipse.
- **...**: Other `ellipse.default` parameters may also be used.

### Details

The summary function is used to obtain the approximate covariance matrix of the fitted parameters.

### Value

A matrix with columns `x` and `y` to outline the confidence region.

### See Also

- `ellipse.default`
- `ellipse.profile`
Examples

# Plot an approximate 95% confidence region for the weight and displacement
# parameters in the Michaelis Menten model
data(Puromycin)
fit <- nls(rate ~ Vm*conc/(K + conc), data = Puromycin, subset = state == "treated",
          start = list(K = 0.05, Vm = 200))
plot(ellipse(fit, which = c("Vm", "K")), type = "l")
params <- fit$m$getPars()
points(params["Vm"], params["K"])

Description

This routine approximates a contour of a function based on the profile of that function.

Usage

## S3 method for class 'profile'
ellipse(x, which = c(1, 2), level = 0.95, t = sqrt(qchisq(level, 2)),
        npoints = 100, ...)

Arguments

x An object of class profile, e.g. from profile.glm in the MASS package.
which Which pair of parameters to use.
level The ellipse.profile function defaults assume that the profiled function is -2
times the log likelihood of a regular model. With this assumption the level
argument specifies the confidence level for an asymptotic confidence region.
t The square root of the value to be contoured.
npoints How many points to use in the ellipse.
... Extra arguments are not used.

Details

This function uses the 4 point approximation to the contour as described in Appendix 6 of Bates
and Watts (1988). It produces the exact contour for quadratic surfaces, and good approximations
for mild deviations from quadratic. If the surface is multimodal, the algorithm is likely to produce
nonsense.

Value

An npoints x 2 matrix with columns having the chosen parameter names, which approximates a
contour of the function that was profiled.
References


See Also

profile, ellipse.nls

Examples

# Plot an approximate 95% confidence region for the Puromycin parameters Vm and K, and overlay the ellipsoidal region

data(Puromycin)
Purboth <- nls(formula = rate ~ ((Vm + delV * (state == "treated")) * conc)/(K + conc), data = Puromycin, start = list(Vm = 160, delV = 40, K = 0.05))
Pur.prof <- profile(Purboth)
plot(ellipse(Pur.prof, which = c("Vm", "K")), type = "l")
lines(ellipse(Purboth, which = c("Vm", "K")), lty = 2)
params <- Purboth$m$getPars()
points(params["Vm"],params["K"])

---

Description

This routine approximates a pairwise confidence region for a glm model.

Usage

## S3 method for class 'profile.glm'
ellipse(x, which = c(1, 2), level = 0.95, t, npoints = 100, dispersion, ...)

Arguments

x An object of class profile.glm.
which Which pair of parameters to use.
level The level argument specifies the confidence level for an asymptotic confidence region.
t The square root of the value to be contoured. By default, this is qchisq(level, 2) for models with fixed dispersion (i.e. binomial and Poisson), and 2 * qf(level, 2, df) for other models, where df is the residual degrees of freedom.
npoints How many points to use in the ellipse.
dispersion If specified, fixed dispersion is assumed, otherwise the dispersion is taken from the model.

... Extra parameters which are not used (for compatibility with the generic).

Details

This function uses the 4 point approximation to the contour as described in Appendix 6 of Bates and Watts (1988). It produces the exact contour for quadratic surfaces, and good approximations for mild deviations from quadratic. If the surface is multimodal, the algorithm is likely to produce nonsense.

Value

An npoints x 2 matrix with columns having the chosen parameter names, which approximates a contour of the function that was profiled.

References


See Also

profile.glm, glm, ellipse.glm

Examples

## MASS has a pairs.profile function that conflicts with ours, so
## do a little trickery here
noMASS <- is.na(match('package:MASS', search()))
if (noMASS) require(MASS)

## Dobson (1990) Page 93: Randomized Controlled Trial:
counts <- c(18, 17, 15, 20, 10, 20, 25, 13, 12)
outcome <- gl(3, 1, 9)
treatment <- gl(3, 3)
glm.D93 <- glm(counts ~ outcome + treatment, family=poisson())

## Plot an approximate 95% confidence region for the two outcome variables
prof.D93 <- profile(glm.D93)
plot(ellipse(prof.D93, which = 2:3), type = 'l')
lines(ellipse(glm.D93, which = 2:3), lty = 2)
params <- glm.D93$coefficients
points(params[2], params[3])

## Clean up our trickery
if (noMASS) detach('package:MASS')
ellipse.profile.nls

Pairwise profile sketch

Description

This routine approximates a pairwise confidence region for a nonlinear regression model.

Usage

```r
## S3 method for class 'profile.nls'
ellipse(x, which = c(1, 2), level = 0.95,
        t = sqrt(2 * qf(level, 2, attr(x, "summary")$df[2])),
        npoints = 100, ...)
```

Arguments

- `x` An object of class `profile.nls`.
- `which` Which pair of parameters to use.
- `level` The `level` argument specifies the confidence level for an asymptotic confidence region.
- `t` The square root of the value to be contoured.
- `npoints` How many points to use in the ellipse.
- `...` Extra parameters which are not used (for compatibility with the generic).

Details

This function uses the 4 point approximation to the contour as described in Appendix 6 of Bates and Watts (1988). It produces the exact contour for quadratic surfaces, and good approximations for mild deviations from quadratic. If the surface is multimodal, the algorithm is likely to produce nonsense.

Value

An `npoints` x 2 matrix with columns having the chosen parameter names, which approximates a contour of the function that was profiled.

References


See Also

`profile`, `ellipse.nls`
Examples

# Plot an approximate 95% confidence region for the Puromycin parameters Vm and K, and overlay the ellipsoidal region
data(Puromycin)
Purboth <- nls(formula = rate ~ ((Vm + delV * (state == "treated")) * conc)/(K + conc), data = Puromycin,
    start = list(Vm = 160, delV = 40, K = 0.05))
Pur.prof <- profile(Purboth)
plot(ellipse(Pur.prof, which = c("Vm", "K")), type = 'l')
lines(ellipse(Purboth, which = c("Vm", "K")), lty = 2)
params <- Purboth$m$getPars()
points(params["Vm"],params["K"])

Description

This function produces pairwise plots of profile traces, profile sketches, and ellipse approximations to confidence intervals.

Usage

pairs_profile(x, labels = c(names(x), "Profile tau"), panel = lines, invert = TRUE,
    plot.tau = TRUE, plot.trace = TRUE, plot.sketch = TRUE,
    plot.ellipse = FALSE, level = 0.95, ...)

# Deprecated generic function. Use graphics::pairs instead.
pairs(x, ...)

Arguments

x An object of class profile, generally the result of the profile() function.
labels The labels to use for each variable. These default to the variable names.
panel The function to use to draw the sketch in each panel.
invert Whether to swap the axes so things look better.
plot.tau Whether to do the profile tau (profile t) plots.
plot.trace Whether to do the profile trace plots.
plot.sketch Whether to do the profile sketch plots.
plot.ellipse Whether to do the ellipse approximations.
level The nominal confidence level for the profile sketches and ellipses.
... Other plotting parameters.
This function implements the plots used in Bates and Watts (1988) for nonlinear regression diagnostics.

Prior to ellipse version 0.5, the pairs_profile function was a profile method for the pairs generic. This caused various conflicts, because graphics also exports a pairs generic, and package MASS exported a profile method for graphics::pairs. As of R version 4.4.0, the MASS method will be in stats instead.

If x is a profile object then pairs_profile(x) will call the function from this package. If you'd rather use the MASS/stats method, then make sure the appropriate package is loaded, and call pairs(x). (Prior to ellipse 0.5, there were complicated rules to determine what pairs(x) would do; those should still work for now, but ellipse::pairs will disappear in a future release.)

Produces a plot on the current device for each pair of variables in the profile object.


See Also

pairs_profile, ellipse_profile, ellipse.nls

Examples

# Plot everything for the Puromycin data
data(Puromycin)
Purboth <- nls(formula = rate ~ ((Vm + delV * (state == "treated")) * conc)/(K + conc), data = Puromycin, start = list(Vm = 160, delV = 40, K = 0.05))

Pur.prof <- profile(Purboth)
pairs_profile(Pur.prof, plot.ellipse = TRUE)

# Show the corresponding plot from MASS/stats:
if (getRversion() < "4.4.0") {
  loadNamespace("MASS")
} else
  loadNamespace("stats")

graphics::pairs(Pur.prof)
### Description

This function plots a correlation matrix using ellipse-shaped glyphs for each entry. The ellipse represents a level curve of the density of a bivariate normal with the matching correlation.

### Usage

```r
plotcorr(corr, outline = TRUE, col = 'grey', numbers = FALSE,
         type = c("full","lower","upper"),
         diag = (type == "full"), bty = "n", axes = FALSE,
         xlab = ",", ylab = "", asp = 1,
         cex.lab = par("cex.lab"), cex = 0.75*par("cex"),
         mar = 0.1 + c(2,2,4,2), ...)  
```

### Arguments

- **corr**: A matrix containing entries between -1 and 1 to be plotted as correlations.
- **outline**: Whether the ellipses should be outlined in the default colour.
- **col**: Which colour(s) to use to fill the ellipses.
- **numbers**: Whether to plot numerical correlations in place of ellipses. If numbers is TRUE, then the correlations will be rounded to a single decimal place and placed on the plot.
- **type**: Character. Plot "full" matrix or just "upper" or "lower" triangular part of it.
- **diag**: Logical. Plot diagonal elements or not.
- **bty, axes, xlab, ylab, asp, cex.lab, cex, mar**: Graphical parameters which will be passed to `plot` when plotting.

### Details

The ellipses being plotted will be tangent to a unit character square, with the shape chosen to match the required correlation. If numbers = FALSE, the col vector will be recycled to colour each of the ellipses; if TRUE, it will be ignored.

### Author(s)

Duncan Murdoch; Gregor Gorjanc suggested the type and diag options.

### References

See Also

ellipse

Examples

```r
save.par <- par(ask = interactive())

# Plot the correlation matrix for the mtcars data full model fit
data(mtcars)
fit <- lm(mpg ~ ., mtcars)
plotcorr(summary(fit, correlation = TRUE)$correlation)

# Plot a second figure with numbers in place of the ellipses
plotcorr(summary(fit, correlation = TRUE)$correlation, numbers = TRUE)

# Colour the ellipses to emphasize the differences. The color range
# is based on RColorBrewer's Reds and Blues (suggested by Gregor Gorjanc)
corr.mtcars <- cor(mtcars)
ord <- order(corr.mtcars[1,])
xc <- corr.mtcars[ord, ord]
colors <- c("#A50F15", "#DE2D26", "#FB6A4A", "#FCAE91", "#FEE5D9", "white",
           "#EFF3FF", "#BDD7E7", "#6BAED6", "#3182BD", "#08519C")
plotcorr(xc, col=colors[5*xc + 6])
plotcorr(xc, col=colors[5*xc + 6], type = "upper")
plotcorr(xc, col=colors[5*xc + 6], type = "lower", diag = TRUE)
par(save.par)
```
Index

* dplot
  ellipse, 3
  ellipse-package, 2
  ellipse.arima0, 4
  ellipse.glm, 5
  ellipse.lm, 6
  ellipse.nls, 7
  ellipse.profile, 8
  ellipse.profile(glm, 9
  ellipse.profile.nls, 11
  pairs_profile, 12

* hplot
  plotcorr, 14

* models
  ellipse.profile, 8
  ellipse.profile(glm, 9
  ellipse.profile.nls, 11

* nonlinear
  ellipse.nls, 7
  pairs_profile, 12

* package
  ellipse-package, 2

* regression
  ellipse.glm, 5
  ellipse.lm, 6
  pairs_profile, 12

* ts
  ellipse.arima0, 4

arima, 2

ellipse, 2, 3, 5, 15
ellipse-deprecated(pairs_profile), 12
ellipse-package, 2
ellipse.arima0, 4, 4
ellipse.default, 6, 7
ellipse.glm, 5, 10
ellipse.lm, 4, 6
ellipse.nls, 4, 7, 9, 11, 13
ellipse.profile, 2, 4, 7, 8, 13

glm, 10
lm, 2
nls, 2
pairs, 13
pairs(pairs_profile), 12
pairs_profile, 2
pairs_profile, 12
plot, 14
plotcorr, 2, 4, 14
profile, 8–11, 13
profile(glm, 8, 9
profile.nls, 11
text, 14