Package: mcprofile (via r-universe)

November 23, 2026

Title Testing Generalized Linear Hypotheses for Generalized Linear Model Parameters by Profile Deviance

Version 1.1-0

Description Calculation of signed root deviance profiles for linear combinations of parameters in a generalized linear model. Multiple tests and simultaneous confidence intervals are provided.

Depends R (>= 4.1.0)

Imports ggplot2 (>= 3.4.0), quadprog, mvtnorm, splines

LazyData yes

Encoding UTF-8

License GPL (>= 2)

URL https://daniel-gerhard.github.io/mcprofile/

VignetteBuilder knitr

Suggests knitr, rmarkdown, multcomp, MASS, testthat (>= 3.0.0)

Config/testthat/edition 3

Config/roxygen2/version 8.1.1

NeedsCompilation no

Author Daniel Gerhard [aut, cre]

Maintainer Daniel Gerhard <00gerhard@gmail.com>

Repository https://cran.r-universe.dev

Date/Publication 2026-11-23 05:10:09 UTC

RemoteUrl https://github.com/cran/mcprofile

RemoteRef HEAD

RemoteSha dfcc7495f1e5d7678106a341b7b0a948799f5813

Contents

aphidlight 2
confint.mcprofile 3
cta 4
exp.mcpCI 4
expit 5
hoa 6
mcprofile 7
mcprofileControl 9
orglm.fit 9
summary.mcprofile 12
toxinLD 13
wald 15

Index **17**

`aphidlight` *Aphid attraction at different light intensities*

Description

The light intensity (mumol/m^2s) of green LED light should be found, which attracts Aphis fabae best. At each of 4 replicates 20 aphids were put in a lightproof box with only one green LED at one end. All aphids that fly to the green light are caught and counted after a period of 5h. This procedure was replicated for 9 increasing light intensities.

Usage

```
aphidlight
```

Format

A data frame with 36 observations on the following 3 variables.

`light` a numeric vector denoting the concentration levels

`black` a numeric vector with the number of aphids remaining in the box.

`green` a numeric vector with the number of attracted aphids

References

Akyazi, G (2009): Zum Einfluss auf Lichtintensitaet und Lichtqualitaet (Hochleistungs-LEDs) auf das Verhalten von Aphis fabae. IPP MSc 19.

confint.mcprofile *Simultaneous Confidence Intervals for Multiple Contrast Profiles*

Description

Calculates simultaneous confidence intervals based on signed root deviance profiles from function mcprofile.

Usage

```
## S3 method for class 'mcprofile'
confint(
  object,
  parm,
  level = 0.95,
  adjust = c("single-step", "none", "bonferroni"),
  alternative = c("two.sided", "less", "greater"),
  ...
)
```

Arguments

object	An object of class mcprofile
parm	Not used; intervals are computed for all contrasts in object.
level	Simultaneous confidence level (1-alpha), default at 0.95
adjust	a character string specifying the adjustment for multiplicity. "single-step" controlling the FWER utilising a multivariate normal- or t-distribution; "none" for comparison-wise error rate; "bonferroni" applying a Bonferroni correction.
alternative	a character string specifying if two- or one-sided confidence intervals should be computed
...	Further arguments, currently ignored.

Value

An object of class mcpCI

See Also

confint.glm, mcprofile, confint.glht

cta *Cell transformation assay dataset*

Description

Balb/c 3T3 cells are treated with different concentrations of a carcinogen. Cells treated with a carcinogen do not stop proliferation. Number of foci (cell accumulations) are counted for 10 replicates per concentration level.

Usage

```
cta
```

Format

A data frame with 80 observations on the following 2 variables.

`conc` a numeric vector denoting the concentration levels

`foci` a numeric vector with the number of foci

References

Thomas C (2008): ECVAM data

exp.mcpCI *exp transformation of Confidence Intervals*

Description

Exponential transformation of confidence interval estimates in mcpCI objects.

Usage

```
## S3 method for class 'mcpCI'
exp(x)
```

Arguments

`x`	An object of class mcpCI

Value

An object of class mcpCI with transformed estimates.

See Also

`exp`, `confint.mcprofile`

Other confidence interval transformations: `expit()`

`expit` *Inverse logit transformation of Confidence Intervals*

Description

Inverse logit transformation of confidence interval estimates in mcpCI objects.

Usage

```
expit(x)

## Default S3 method:
expit(x)

## S3 method for class 'mcpCI'
expit(x)
```

Arguments

`x`	An object of class mcpCI, or a numeric vector, matrix or data.frame for the default method.

Value

An object of class mcpCI with transformed estimates. The default method returns `1/(1 + exp(-x))`.

See Also

`exp`, `confint.mcprofile`

Other confidence interval transformations: `exp.mcpCI()`

Examples

```
## proportion of green aphids at each light level
aphidlight$lightf <- factor(aphidlight$light)
fm <- glm(cbind(green, black) ~ lightf - 1, data = aphidlight,
          family = binomial(link = "logit"))

## one contrast per light level: the logit of each group's proportion
CM <- diag(nlevels(aphidlight$lightf))
rownames(CM) <- paste("light", levels(aphidlight$lightf))

mc <- mcprofile(fm, CM)
(ci <- confint(mc))

## simultaneous confidence intervals on the probability scale
(pci <- expit(ci))
plot(pci)
```

hoa *Higher order asymptotics using the modified likelihood root*

Description

Transforms a signed root deviance profile to a modified likelihood root profile.

Usage

```
hoa(object, maxstat = 10)
```

Arguments

object	An object of class mcprofile
maxstat	Limits the statistic to a maximum absolute value (default=10)

Value

An object of class mcprofile with a hoa profile in the srdp slot.

See Also

mcprofile

Examples

```
#######################################
## cell transformation assay example ##
#######################################

str(cta)
## change class of cta$conc into factor
cta$concf <- factor(cta$conc, levels=unique(cta$conc))

library(ggplot2)
ggplot(cta, aes(y=foci, x=concf)) +
  geom_boxplot() +
  geom_dotplot(binaxis = "y", stackdir = "center", binwidth = 0.2) +
  xlab("concentration")


# glm fit assuming a Poisson distribution for foci counts
# parameter estimation on the log link
# removing the intercept
fm <- glm(foci ~ concf-1, data=cta, family=poisson(link="log"))

### Comparing each dose to the control by Dunnett-type comparisons
# Constructing contrast matrix, the same as
# multcomp::contrMat(table(cta$concf), type = "Dunnett")
```

```
lev <- levels(cta$concf)
CM <- cbind(-1, diag(length(lev) - 1))
dimnames(CM) <- list(paste(lev[-1], "-", lev[1]), lev)

# calculating signed root deviance profiles
(dmcp <- mcprofile(fm, CM))
# computing profiles for the modified likelihood root
hp <- hoa(dmcp)

plot(hp)

# comparing confidence intervals
confint(hp)
confint(dmcp)
```

`mcprofile` *Construction of Multiple Contrast Profiles*

Description

Calculates signed root deviance profiles given a `glm` or `lm` object. The profiled parameters of interest are defined by providing a contrast matrix.

Usage

```
mcprofile(object, CM, control = mcprofileControl(), grid = NULL)

## S3 method for class 'glm'
mcprofile(object, CM, control = mcprofileControl(), grid = NULL)

## S3 method for class 'lm'
mcprofile(object, CM, control = mcprofileControl(), grid = NULL)
```

Arguments

`object`	An object of class `glm` or `lm`
`CM`	A contrast matrix for the definition of parameter linear combinations (`CM %*% coefficients(object)`). The number of columns should be equal to the number of estimated parameters. Providing row names is recommendable.
`control`	A list with control arguments. See `mcprofileControl`.
`grid`	A matrix or list with profile support coordinates. Each column of the matrix or slot in a list corresponds to a row in the contrast matrix, each row of the grid matrix or element of a numeric vector in each list slot corresponds to a candidate of the contrast parameter. If NULL (default), a grid is found automatically similar to function `profile.glm`.

Details

The profiles are calculated separately for each row of the contrast matrix. The profiles are calculated by constrained IRWLS optimization, implemented in function `orglm`, using the quadratic programming algorithm of package `quadprog`. Models with aliased (`NA`) coefficients are not supported; refit the model without the redundant terms.

Value

An object of class mcprofile. The slot `srdp` contains the profiled signed root deviance statistics. The `optpar` slot contains a matrix with profiled parameter estimates.

See Also

`profile.glm`, `glht`, `contrMat`, `confint.mcprofile`, `summary.mcprofile`, `solve.QP`

Examples

```
#######################################
## cell transformation assay example ##
#######################################

str(cta)
## change class of cta$conc into factor
cta$concf <- factor(cta$conc, levels=unique(cta$conc))

library(ggplot2)
ggplot(cta, aes(y=foci, x=concf)) +
  geom_boxplot() +
  geom_dotplot(binaxis = "y", stackdir = "center", binwidth = 0.2) +
  xlab("concentration")


# glm fit assuming a Poisson distribution for foci counts
# parameter estimation on the log link
# removing the intercept
fm <- glm(foci ~ concf-1, data=cta, family=poisson(link="log"))

### Comparing each dose to the control by Dunnett-type comparisons
# Constructing contrast matrix, the same as
# multcomp::contrMat(table(cta$concf), type = "Dunnett")
lev <- levels(cta$concf)
CM <- cbind(-1, diag(length(lev) - 1))
dimnames(CM) <- list(paste(lev[-1], "-", lev[1]), lev)

# calculating signed root deviance profiles
(dmcp <- mcprofile(fm, CM))
# plot profiles
plot(dmcp)
# confidence intervals
(ci <- confint(dmcp))
plot(ci)
```

mcprofileControl *mcprofile Control Arguments*

Description

Control arguments for the mcprofile function

Usage

```
mcprofileControl(maxsteps = 10, alpha = 0.01, del = function(zmax) zmax/5)
```

Arguments

maxsteps	Maximum number of points to be used for profiling each parameter.
alpha	Highest significance level allowed for the profile t-statistics (Bonferroni adjusted)
del	Suggested change on the scale of the profile t-statistics. Default value chosen to allow profiling at about 10 parameter values.

See Also

mcprofile

orglm.fit *Fitting Order-Restricted Generalized Linear Models*

Description

orglm.fit is used to fit generalized linear models with restrictions on the parameters, specified by giving a description of the linear predictor, a description of the error distribution, and a description of a matrix with linear constraints. The quadprog package is used to apply linear constraints on the parameter vector.

Usage

```
orglm.fit(
  x,
  y,
  weights = rep(1, nobs),
  start = NULL,
  etastart = NULL,
  mustart = NULL,
  offset = rep(0, nobs),
  family = gaussian(),
  control = list(),
```

```
  intercept = TRUE,
  constr,
  rhs,
  nec
)
```

Arguments

x	is a design matrix of dimension n * p
y	is a vector of observations of length n
weights	an optional vector of 'prior weights' to be used in the fitting process. Should be NULL or a numeric vector.
start	starting values for the parameters in the linear predictor.
etastart	starting values for the linear predictor.
mustart	starting values for the vector of means.
offset	this can be used to specify an *a priori* known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases. One or more offset terms can be included in the formula instead or as well, and if more than one is specified their sum is used. See model.offset.
family	a description of the error distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. (See family for details of family functions.)
control	a list of parameters for controlling the fitting process. For orglm.fit this is passed to glm.control.
intercept	logical. Should an intercept be included in the *null* model?
constr	a matrix with linear constraints. The columns of this matrix should correspond to the columns of the design matrix.
rhs	right hand side of the linear constraint formulation. A numeric vector with a length corresponding to the rows of constr.
nec	Number of equality constraints. The first nec constraints defined in constr are treated as equality constraints; the remaining ones are inequality constraints.

Details

Non-NULL weights can be used to indicate that different observations have different dispersions (with the values in weights being inversely proportional to the dispersions); or equivalently, when the elements of weights are positive integers w_i, that each response y_i is the mean of w_i unit-weight observations. For a binomial GLM prior weights are used to give the number of trials when the response is the proportion of successes: they would rarely be used for a Poisson GLM. If more than one of etastart, start and mustart is specified, the first in the list will be used. It is often advisable to supply starting values for a quasi family, and also for families with unusual links such as gaussian("log"). For the background to warning messages about 'fitted probabilities numerically 0 or 1 occurred' for binomial GLMs, see Venables & Ripley (2002, pp. 197–8).

Value

An object of class "glm" is a list containing at least the following components:

coefficients a named vector of coefficients

residuals the *working* residuals, that is the residuals in the final iteration of the IWLS fit. Since cases with zero weights are omitted, their working residuals are NA.

fitted.values the fitted mean values, obtained by transforming the linear predictors by the inverse of the link function.

rank the numeric rank of the fitted linear model.

family the family object used.

linear.predictors the linear fit on link scale.

deviance up to a constant, minus twice the maximized log-likelihood. Where sensible, the constant is chosen so that a saturated model has deviance zero.

null.deviance The deviance for the null model, comparable with deviance. The null model will include the offset, and an intercept if there is one in the model. Note that this will be incorrect if the link function depends on the data other than through the fitted mean: specify a zero offset to force a correct calculation.

iter the number of iterations of IWLS used.

weights the *working* weights, that is the weights in the final iteration of the IWLS fit.

prior.weights the weights initially supplied, a vector of 1s if none were.

df.residual the residual degrees of freedom of the unconstrained model.

df.null the residual degrees of freedom for the null model.

y if requested (the default) the y vector used. (It is a vector even for a binomial model.)

converged logical. Was the IWLS algorithm judged to have converged?

boundary logical. Is the fitted value on the boundary of the attainable values?

Author(s)

Modification of the original glm.fit by Daniel Gerhard. The original R implementation of glm was written by Simon Davies working for Ross Ihaka at the University of Auckland, but has since been extensively re-written by members of the R Core team. The design was inspired by the S function of the same name described in Hastie & Pregibon (1992).

References

- Dobson, A. J. (1990) *An Introduction to Generalized Linear Models.* London: Chapman and Hall.
- Hastie, T. J. and Pregibon, D. (1992) *Generalized linear models.* Chapter 6 of *Statistical Models in S* eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.
- McCullagh P. and Nelder, J. A. (1989) *Generalized Linear Models.* London: Chapman and Hall.
- Venables, W. N. and Ripley, B. D. (2002) *Modern Applied Statistics with S.* New York: Springer.

See Also

glm, solve.QP

---

summary.mcprofile *Multiple Testing of General Hypotheses*

Description

Multiple contrast testing based on signed root deviance profiles.

Usage

```
## S3 method for class 'mcprofile'
summary(
  object,
  margin = 0,
 adjust = c("single-step", "holm", "hochberg", "hommel", "bonferroni", "BH", "BY",
    "fdr", "none"),
  alternative = c("two.sided", "less", "greater"),
  ...
)
```

Arguments

object	an object of class mcprofile
margin	test margin, specifying the right hand side of the hypotheses.
adjust	a character string specifying the adjustment for multiplicity. "single-step" controlling the FWER utilizing a multivariate normal- or t-distribution; "none" for comparison-wise error rate, or any other method provided by p.adjust.
alternative	a character string specifying the alternative hypothesis.
...	Further arguments, currently ignored.

Value

An object of class mcpSummary

See Also

mcprofile, summary.glht

toxinLD *Identifying the lethal dose of a crop protection product.*

Description

Increasing dose levels of a toxin, used as a pesticide for crop protection, is applied to non-target species. The lethal dose should be identified in this experiment. The dataset represents simulated data based on a real experiment.

Usage

```
toxinLD
```

Format

A data frame with 6 observations on the following 3 variables.

`dose` a numeric vector denoting the toxin concentration levels

`dead` a numeric vector with the number of dead insects.

`alive` a numeric vector with the number of surviving insects.

Examples

```
str(toxinLD)

################################################
# logistic regression on the logarithmic dose #
################################################

toxinLD$logdose <- log(toxinLD$dose)
fm <- glm(cbind(dead, alive) ~ logdose, data=toxinLD, family=binomial(link="logit"))

#############
# profiling #
#############

# contrast matrix
pdose <- seq(-1,2.3, length=7)
CM <- model.matrix(~ pdose)

# user defined grid to construct profiles
mcpgrid <- matrix(seq(-11,8,length=15), nrow=15, ncol=nrow(CM))
mc <- mcprofile(fm, CM, grid=mcpgrid)

###################################
## confidence interval calculation #
###################################

# srdp profile
```

```
ci <- confint(mc)
ppdat <- data.frame(logdose=pdose)
ppdat$estimate <- fm$family$linkinv(ci$estimate$Estimate)
ppdat$lower <- fm$family$linkinv(ci$confint$lower)
ppdat$upper <- fm$family$linkinv(ci$confint$upper)
ppdat$method <- "profile"

# wald profile
wci <- confint(wald(mc))
wpdat <- ppdat
wpdat$estimate <- fm$family$linkinv(wci$estimate$Estimate)
wpdat$lower <- fm$family$linkinv(wci$confint$lower)
wpdat$upper <- fm$family$linkinv(wci$confint$upper)
wpdat$method <- "wald"

# higher order approximation
hci <- confint(hoa(mc))
hpdat <- ppdat
hpdat$estimate <- fm$family$linkinv(hci$estimate$Estimate)
hpdat$lower <- fm$family$linkinv(hci$confint$lower)
hpdat$upper <- fm$family$linkinv(hci$confint$upper)
hpdat$method <- "hoa"

# combine results
pdat <- rbind(ppdat, wpdat, hpdat)


####################################
# estimating the lethal dose LD(25) #
####################################

ld <- 0.25
pspf <- splinefun(ppdat$upper, pdose)
pll <- pspf(ld)
wspf <- splinefun(wpdat$upper, pdose)
wll <- wspf(ld)
hspf <- splinefun(hpdat$upper, pdose)
hll <- hspf(ld)

ldest <- data.frame(limit=c(pll, wll, hll), method=c("profile","wald", "hoa"))

################################
# plot of intervals and LD(25) #
################################

library(ggplot2)
ggplot(toxinLD, aes(x=logdose, y=dead/(dead+alive))) +
  geom_ribbon(data=pdat, aes(y=estimate, ymin=lower, ymax=upper,
                             fill=method, colour=method, linetype=method),
              alpha=0.1, linewidth=0.95) +
  geom_line(data=pdat, aes(y=estimate, linetype=method), linewidth=0.95) +
  geom_point(size=3) +
  geom_hline(yintercept=ld, linetype=2) +
```

```
  geom_segment(data=ldest, aes(x=limit, xend=limit, y=0.25, yend=-0.05,
                               linetype=method), linewidth=0.6, colour="grey2") +
  ylab("Mortality rate")
```

---

wald *Calculate Wald-Profiles*

Description

Transforms a signed root deviance profile of a mcprofile object into a profile of Wald-type statistics

Usage

```
wald(object)
```

Arguments

object	An object of class mcprofile

Value

An object of class mcprofile with a wald profile in the srdp slot.

See Also

mcprofile

Examples

```
######################################
## cell transformation assay example ##
######################################

str(cta)
## change class of cta$conc into factor
cta$concf <- factor(cta$conc, levels=unique(cta$conc))

library(ggplot2)
ggplot(cta, aes(y=foci, x=concf)) +
  geom_boxplot() +
  geom_dotplot(binaxis = "y", stackdir = "center", binwidth = 0.2) +
  xlab("concentration")


# glm fit assuming a Poisson distribution for foci counts
# parameter estimation on the log link
# removing the intercept
fm <- glm(foci ~ concf-1, data=cta, family=poisson(link="log"))

### Comparing each dose to the control by Dunnett-type comparisons
```

```
# Constructing contrast matrix, the same as
# multcomp::contrMat(table(cta$concf), type = "Dunnett")
lev <- levels(cta$concf)
CM <- cbind(-1, diag(length(lev) - 1))
dimnames(CM) <- list(paste(lev[-1], "-", lev[1]), lev)

# calculating signed root deviance profiles
(dmcp <- mcprofile(fm, CM))
# computing profiles for the modified likelihood root
wp <- wald(dmcp)

plot(wp)

# comparing confidence intervals
confint(wp)
confint(dmcp)
```

Index

∗ **confidence interval transformations**
  exp.mcpCI, 4
  expit, 5
∗ **datasets**
  aphidlight, 2
  cta, 4
  toxinLD, 13
∗ **htest**
  confint.mcprofile, 3
  summary.mcprofile, 12
∗ **mcprofile**
  mcprofile, 7
∗ **misc**
  exp.mcpCI, 4
  expit, 5
  hoa, 6
  mcprofile, 7
  mcprofileControl, 9
  wald, 15
∗ **models**
  orglm.fit, 9

aphidlight, 2

confint.glht, *3*
confint.glm, *3*
confint.mcprofile, 3, *4, 5*, *8*
contrMat, *8*
cta, 4

exp, *4, 5*
exp.mcpCI, 4
exp.mcpCI(), *5*
expit, 5
expit(), *4*

family, *10, 11*

glht, *8*
glm, *7*, *12*
glm.control, *10*

hoa, 6

lm, *7*

mcprofile, *3*, *6*, 7, *9*, *12*, *15*
mcprofileControl, *7*, 9
model.offset, *10*

offset, *10*
orglm.fit, 9

p.adjust, *12*
profile.glm, *7, 8*

quasi, *10*

solve.QP, *8*, *12*
summary.glht, *12*
summary.mcprofile, *8*, 12

toxinLD, 13

wald, 15