Package: hnp (via r-universe)

September 12, 2024

Type Package

Title Half-Normal Plots with Simulation Envelopes

Version 1.2-6

Date 2018-05-21

Author Rafael de Andrade Moral [aut, cre], John Hinde [aut], Clarice Garcia Borges Demetrio [aut]

Maintainer Rafael de Andrade Moral <rafael_moral@yahoo.com.br>

Depends R (>= 3.0.0), MASS (>= 7.3-35), methods, graphics, stats

Suggests lme4, gamlss, gamlss.dist, pscl, nnet, aods3, VGAM, glmmADMB, latticeExtra

Additional_repositories http://glmmadmb.r-forge.r-project.org/repos

Description Generates (half-)normal plots with simulation envelopes using different diagnostics from a range of different fitted models. A few example datasets are included.

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2018-05-21 15:27:55 UTC

Contents

---

hnp-package *Half-Normal Plots with Simulation Envelopes*

Description

Generates (half-)normal plots with simulation envelopes for common diagnostics from fitted models for a range of model classes. The function `hnp()` is written so that it is relatively easy to extend it to new model classes and different diagnostics that are not already implemented. A few example datasets are included. The main functions are `hnp` and `plot.hnp`

Details

Package:	hnp
Type:	Package
Version:	1.2-6
Date:	2018-05-21
License:	GPL version 2 or newer

Package includes two functions, `hnp` and `plot.hnp`, as well as seven data sets extracted from Demétrio et al (2014). Explanation on how to produce the half-normal plots with simulation envelopes can be found in Demétrio and Hinde (1997) and on the `hnp` function documentation. Data sets are based on entomology studies and the focus is on overdispersed data. Details on overdispersion models can be found in Hinde and Demétrio (1998). Special thanks to FAPESP and CNPq for funding.

Author(s)

Rafael A. Moral, John Hinde and Clarice G. B. Demétrio

Maintainer: Rafael Moral <rafael_moral@yahoo.com.br>

References

Moral, R. A., Hinde, J. and Demétrio, C. G. B. (2017) Half-normal plots and overdispersed models in R: the hnp package. Journal of Statistical Software 81(10):1-23.

Demétrio, C. G. B. and Hinde, J. (1997) Half-normal plots and overdispersion. GLIM Newsletter 27:19-26.

Hinde, J. and Demétrio, C. G. B. (1998) Overdispersion: models and estimation. Computational Statistics and Data Analysis 27:151-170.

Demétrio, C. G. B., Hinde, J. and Moral, R. A. (2014) Models for overdispersed data in entomology. In Godoy, W. A. C. and Ferreira, C. P. (Eds.) Ecological modelling applied to entomology. Springer.

cbb *Coffee berry borer trapping data*

Description

Data on counts of coffee berry borer obtained using different traps through time.

Usage

```
data(cbb)
```

Format

A data frame with 288 observations on the following 4 variables.

`week`	numeric	week of observation (1 to 24)
`block`	factor	levels `I` `II` `III` `IV`
`trap`	factor	levels `CV` `F` `SF`
`count`	numeric	number of observed insects

Details

The coffee berry borer is a major pest of commercial coffee. The insect directly attacks the coffee fruit in development causing severe losses in bean production and quality. This data set was obtained in an experiment conducted by Mota (2013), where three types of traps (`SF`, `F`, `CV`) were randomized in each of four equidistant lines (blocks) of a coffee field. Each week, over a 24 week period, the insects were removed from the traps and counted.

Source

Demétrio, C. G. B., Hinde, J. and Moral, R. A. (2014) Models for overdispersed data in entomology. In Godoy, W. A. C. and Ferreira, C. P. (Eds.) Ecological modelling applied to entomology. Springer.

References

Mota, L. H. C. (2013) Desenvolvimento de armadilha de auto-inoculacao para o controlde de Hypothenemus hampei (Ferrari, 1867) (Coleoptera: Curculionidae) com Beauveria bassiana (Bals.) Vuil (Ascomycota: Hypocreales) em tecido sinetico. Master's dissertation, ESALQ-USP

Examples

```
data(cbb)
# exploratory plot
require(latticeExtra)
trellis.par.set(strip.background=list(col="lightgrey"))
useOuterStrips(xyplot(count ~ week | block + trap, data=cbb,
  layout=c(3,1),type="l", col=1, xlab="Week", ylab="Insect counts"))
```

```
# Poisson fit
model1 <- glm(count ~ block + trap*factor(week),
              data=cbb, family=poisson)
anova(model1, test="Chisq")
sum(resid(model1, ty="pearson")^2)
summary(model1)
hnp(model1, sim=19, conf=1)

## Not run:
hnp(model1) # default call

## End(Not run)

# Quasi-Poisson fit
model2 <- glm(count ~ block + trap*factor(week), data=cbb,
              family=quasipoisson)
anova(model2, test="F")
summary(model2)
hnp(model2, sim=19, conf=1)

## Not run:
hnp(model2) # default call

## End(Not run)

## for discussion on the analysis of this data set,
## see Demetrio et al. (2014)
```

`chryso` Chrysoperla externa *mortality data*

Description

Mortality of the predator *Chrysoperla externa* on different doses of lime sulphur, a substance used to control pests on trees.

Usage

```
data(chryso)
```

Format

A data frame with 24 observations on the following 4 variables.

`dead`	numeric	count of dead C. externa specimens
`alive`	numeric	count of alive C. externa specimens
`conc`	numeric	lime sulphur concentration
`log.conc`	numeric	natural logarithm of lime sulphur concentration

Details

The neuropteran *Chrysoperla externa* is a predator that acts as a natural enemy of the brown citrus aphid, *Toxoptera citricida*, which is among the most important citrus pests worldwide. A possible strategy to control *T. citricida* populations would be to use a substance called lime sulphur and the *C. externa* predator in combination, which may be beneficial as long as the lime sulphur has less effect on the predator than the prey. To explore this, Battel (2012) conducted an experiment with first-instar larvae of *Chrysoperla externa* exposed to different levels of lime sulphur. Specifically, twenty-four Orange Jessamine (*Murraya paniculata*) plants were sprayed with different concentrations (conc) of lime sulphur and up to seven first-instar larvae were placed on each plant. The experiment was set up in a completely randomized design with four treatments: lime sulphur concentrations at 0ppm (water control), 60ppm, 600ppm, and 6000ppm. The plants were observed until the predators reached the second instar and the number of larvae that died on each plant was recorded.

Source

Demétrio, C. G. B., Hinde, J. and Moral, R. A. (2014) Models for overdispersed data in entomology. In Godoy, W. A. C. and Ferreira, C. P. (Eds.) Ecological modelling applied to entomology. Springer.

References

Battel, A. P. M. B. (2012) Dinamica de predacao e resposta funcional em *Chrysoperla externa* (*Neuroptera: Chrysopidae*) sobre *Toxoptera citricida* (*Hemiptera: Aphididae*) aplicada a citricultura organica. Master's dissertation, ESALQ-USP

Examples

```
data(chryso)

# fit model using conc levels both on log-scale and
# as a factor to produce simple analysis of deviance
model <- glm(cbind(dead, alive) ~  log.conc + factor(conc),
             family=binomial, data=chryso)
anova(model, test="Chisq")
# test adequacy of factor model using deviance and X2
1-pchisq(deviance(model), df.residual(model))
(X2 <- sum(residuals(model, type="pearson")^2))
1-pchisq(X2, df.residual(model))

model1 <- glm(cbind(dead, alive) ~ log.conc,
              family=binomial, data=chryso)
par(mfrow=c(1,2), cex=1.4)
with(chryso, plot(jitter(log.conc), dead/(dead+alive),
                  ylab="Proportion dead", xlab="log(conc+1)"))
x <- seq(0, 8.7, .1)
pr <- predict(model1, data.frame(log.conc=x), ty="response")
lines(x, pr)
# half-normal plot
hnp(model1, xlab="Half-normal scores",
    ylab="Deviance residuals", pch=4)
```

```
require(MASS)
dose.p(model1, p=.10)

logLC10 <- dose.p(model1, p=.10)
LC10 <- exp(logLC10[[1]]) - 1

#95% CI on log-dose scale using transformation
c(logLC10[1]-2*attr(logLC10,'SE'),
  logLC10[1]+2*attr(logLC10,'SE'))

#95% CI on dose scale using transformation
c(exp(logLC10[1]-2*attr(logLC10,'SE'))-1,
  exp(logLC10[1]+2*attr(logLC10,'SE'))-1)

## for discussion on the analysis of this data set,
## see Demetrio et al. (2014)
```

`corn` *Corn damage data*

Description

Corn grain damage by the maize weevil, a major pest of stored maize worldwide.

Usage

```
data(corn)
```

Format

A data frame with 40 observations on the following 3 variables.

`extract`	factor	with levels `leaf`, `branch`, `seed` and `control`
`m`	numeric	total number of corn grains
`y`	numeric	number of damaged corn grains

Details

A major pest of stored maize in Brazil is *Sitophilus zeamais*. In an experiment to assess the insecticide action of organic extracts of *Annona mucosa* (*Annonaceae*), Petri dishes containing 10g of corn were treated with extracts prepared with different parts of the plant (seeds, leaves and branches) at a concentration of 1500mg/kg or just water (control), using a completely randomized design with 10 replicates. Then 20 Sitophilus zeamais adults were placed in each Petri dish and, after 60 days, the numbers of damaged and undamaged corn grains were counted, see Ribeiro et al (2013).

Source

Demétrio, C. G. B., Hinde, J. and Moral, R. A. (2014) Models for overdispersed data in entomology. In Godoy, W. A. C. and Ferreira, C. P. (Eds.) Ecological modelling applied to entomology. Springer.

References

Moral, R. A., Hinde, J. and Demétrio, C. G. B. (2017) Half-normal plots and overdispersed models in R: the hnp package. Journal of Statistical Software 81(10):1-23.

Ribeiro, L. P., Vendramin, J. D., Bicalho, K. U., Andrade, M. S., Fernandes, J. B., Moral, R. A., Demétrio, C. G. B. (2013) *Annona mucosa Jacq.* (*Annonaceae*): A promising source of bioactive compounds against *Sitophilus zeamais Mots.* (*Coleoptera: Curculionidae*). J Stored Prod Res 55:6-14

Examples

```
data(corn)

# Binomial fit
model1 <- glm(cbind(y, m-y) ~ extract, family=binomial,
              data=corn)
anova(model1, test="Chisq")
hnp(model1, pch=4, main="Binomial: Logit",
    xlab="Half-normal scores", ylab="Deviance residuals")

# Quasi-binomial fit
model2 <- glm(cbind(y, m-y) ~ extract, family=quasibinomial,
              data=corn)
anova(model2, test="F")
summary(model2)$dispersion # estimated phi

# half-normal plots
par(mfrow=c(1,2),cex=1.4, cex.main=0.9, pty='s')
hnp(model1, pch=4, main="(a) Binomial; Logit",
    xlab="Half-normal scores", ylab="Deviance residuals")
hnp(model2, pch=4, main="(b) Quasibinomial; Logit",
    xlab="Half-normal scores", ylab="Deviance residuals")

anova(model1, test="Chisq") # binomial model
anova(model2, test="F") # quasi-binomial model
summary(model1) # binomial model
summary(model2) # quasi-binomial model

# now with factor level parameterisation
summary(update(model1,.~.-1))
summary(update(model2,.~.-1))

## for discussion on the analysis of this data set,
## see Demetrio et al. (2014)
```

`fungi` Diaphorina citri *mortality data*

Description

Mortality of the Citrus psyllid, *Diaphorina citri*, a major pest of Citrus worldwide, when exposed to different concentrations of two fungi species, *Beauveria bassiana* and *Isaria fumosorosea*.

Usage

```
data(fungi)
```

Format

A data frame with 30 observations on the following 5 variables.

`y`	numeric	number of dead insects
`m`	numeric	total number of insects
`conc`	numeric	fungi concentration (in conidia/ml)
`lconc`	numeric	natural logarithm of fungi concentration
`species`	factor	levels `isaria` and `beauveria`, fungi species

Details

The Citrus psyllid *Diaphorina citri* is a vector of Huanglongbing, known as greening disease. An alternative to chemical control is to use solutions of fungi conidia as a biological control strategy. D'Alessandro (2014) conducted a completely randomized experiment to assess how different conidia concentrations (10^4, 10^5, 10^6, 10^7 and 10^8 conidia/ml) of two fungi species, *Beauveria bassiana* and *Isaria fumosorosea*, infected *D. citri* adults. Each experimental unit consisted of around 20 *D. citri* adults, which were placed on *Citrus limonia* plants. The insects were pulverized with the solutions and after 10 days the number of dead insects and dead insects due to fungus infection were observed. Note that in this case the conidia concentrations are obtained in successive dilutions and therefore small variations in the number of conidia per ml may contribute additional variability to the data. Such additional variability may be accounted for in the model by including an additive random effect in the linear predictor.

Source

Demétrio, C. G. B., Hinde, J. and Moral, R. A. (2014) Models for overdispersed data in entomology. In Godoy, W. A. C. and Ferreira, C. P. (Eds.) Ecological modelling applied to entomology. Springer.

References

D'Alessandro (2014) Unpublished data, private communication.

Examples

```
data(fungi)

# Binomial fit
model1 <- glm(cbind(y, m-y) ~ lconc*species,
              family=binomial, data=fungi)
anova(model1, test="Chisq")
sum(resid(model1, ty="pearson")^2)
```

```
1 - pchisq(sum(resid(model1, ty="pearson")^2), 20)
hnp(model1)

# Quasi-binomial fit
model2 <- glm(cbind(y, m-y) ~ lconc*species,
              family=quasibinomial, data=fungi)
anova(model2, test="F")
hnp(model2)

## Not run:
# Logistic-normal fit
require(lme4)
fungi$ind <- factor(1:nrow(fungi))
model3 <- glmer(cbind(y, m-y) ~ lconc*species + (1|ind),
                family=binomial, data=fungi)
summary(model3)
hnp(model3)

## End(Not run)

## for discussion on the analysis of this data set,
## see Demetrio et al. (2014)
```

---

hnp *Half-Normal Plots with Simulation Envelopes*

Description

Produces a (half-)normal plot from a fitted model object for a range of different models. Extendable to non-implemented model classes.

Usage

```
hnp(object, sim = 99, conf = 0.95, resid.type, maxit,
    halfnormal = T, scale = F, plot.sim = T, verb.sim = F,
    warn = F, how.many.out = F, print.on = F, paint.out = F,
    col.paint.out, newclass = F, diagfun, simfun, fitfun, ...)
```

Arguments

`object`	fitted model object or numeric vector.
`sim`	number of simulations used to compute envelope. Default is 99.
`conf`	confidence level of the simulated envelope. Default is 0.95.
`resid.type`	type of residuals to be used; must be one of "deviance", "pearson", "response", "working", "simple", "student", or "standard". Not all model type and residual type combinations are allowed. Defaults are "student" for `aov` and `lm` objects, "deviance" for `glm`, `glm.nb`, `lmer`, `glmer` and `aodml` objects, "simple" for `gamlss` objects, "response" for `glmmadmb` and `vglm` objects, "pearson" for `zeroinfl` and `hurdle` objects.

maxit	maximum number of iterations of the estimation algorithm. Defaults are 25 for glm, glm.nb, gamlss and vglm objects, 300 for glmmadmb, lmer and glmer objects, 3000 for aodml objects, 10000 for zeroinfl and hurdle objects.
halfnormal	logical. If TRUE, a half-normal plot is produced. If FALSE, a normal plot is produced. Default is TRUE.
scale	logical. If TRUE and if object is a numeric vector, simulates from a normal distribution with mean and variance estimated from object. If FALSE, uses a standard normal distribution to simulate from. Default is FALSE.
plot.sim	logical. Should the (half-)normal plot be plotted? Default is TRUE.
verb.sim	logical. If TRUE, prints each step of the simulation procedure. Default is FALSE.
warn	logical. If TRUE, shows warning messages in the simulation process. Default is FALSE.
how.many.out	logical. If TRUE, the number of points out of the envelope is printed. Default is FALSE.
print.on	logical. If TRUE, the number of points out of the envelope is printed on the plot. Default is FALSE.
paint.out	logical. If TRUE, points out of the simulation envelope are plotted in a different color. Default is FALSE.
col.paint.out	If paint.out=TRUE, sets the color of points out of the envelope. Default is "red".
newclass	logical. If TRUE, use diagfun, simfun, and fitfun to extract diagnostics (typically residuals), generate simulated data using fitted model parameters, and fit the desired model. Default is FALSE.
diagfun	user-defined function used to obtain the diagnostic measures from the fitted model object (only used when newclass=TRUE). Default is resid.
simfun	user-defined function used to simulate a random sample from the model estimated parameters (only used when newclass=TRUE).
fitfun	user-defined function used to re-fit the model to simulated data (only used when newclass=TRUE).
...	extra graphical arguments passed to plot.hnp.

Details

A relatively easy way to assess goodness-of-fit of a fitted model is to use (half-)normal plots of a model diagnostic, e.g., different types of residuals, Cook's distance, leverage. These plots are obtained by plotting the ordered absolute values of a model diagnostic versus the expected order statistic of a half-normal distribution,

$$\Phi^{-1}(\frac{i+n-1/8}{2*n+1/2})$$

(for a half-normal plot) or the normal distribution,

$$\Phi^{-1}(\frac{i+3/8}{n+1/4})$$

(for a normal plot).

Atkinson (1985) proposed the addition of a simulated envelope, which is such that under the correct model the plot for the observed data is likely to fall within the envelope. The objective is not to provide a region of acceptance, but some sort of guidance to what kind of shape to expect.

Obtaining the simulated envelope is simple and consists of (1) fitting a model; (2) extracting model diagnostics and calculating sorted absolute values; (3) simulating 99 (or more) response variables using the same model matrix, error distribution and fitted parameters; (4) fitting the same model to each simulated response variable and obtaining the same model diagnostics, again sorted absolute values; (5) computing the desired percentiles (e.g., 2.5 and 97.5) at each value of the expected order statistic to form the envelope.

This function handles different model classes and more will be implemented as time goes by. So far, the following models are included:

Continuous data:

Normal:	functions `lm`, `aov` and `glm` with `family=gaussian`
Gamma:	function `glm` with `family=Gamma`
Inverse gaussian:	function `glm` with `family=inverse.gaussian`

Proportion data:

Binomial:	function `glm` with `family=binomial`
Quasi-binomial:	function `glm` with `family=quasibinomial`
Beta-binomial:	package `VGAM` - function `vglm`, with `family=betabinomial`; package `aods3` - function `aodml`, with `family="bb"`; package `gamlss` - function `gamlss`, with `family=BB`; package `glmmADMB` - function `glmmadmb`, with `family="betabinomial"`
Zero-inflated binomial:	package `VGAM` - function `vglm`, with `family=zibinomial`; package `gamlss` - function `gamlss`, with `family=ZIBI`; package `glmmADMB` - function `glmmadmb`, with `family="binomial"` and `zeroInfl=TRUE`
Zero-inflated beta-binomial:	package `gamlss` - function `gamlss`, with `family=ZIBB`; package `glmmADMB` - function `glmmadmb`, with `family="betabinomial"` and `zeroInfl=TRUE`
Multinomial:	package `nnet` - function `multinom`

Count data:

Poisson:	function `glm` with `family=poisson`

Quasi-Poisson:	function glm with family=quasipoisson
Negative binomial:	package MASS - function glm.nb; package aods3 - function aodml, with family="nb" and phi.scale="inverse"
Zero-inflated Poisson:	package pscl - function zeroinfl, with dist="poisson"
Zero-inflated negative binomial:	package pscl - function zeroinfl, with dist="negbin"
Hurdle Poisson:	package pscl - function hurdle, with dist="poisson"
Hurdle negative binomial:	package pscl - function hurdle, with dist="negbin"

Mixed models:

Linear mixed models:	package lme4, function lmer
Generalized linear mixed models:	package lme4, function glmer with family=poisson or binomial

Users can also use a numeric vector as object and hnp will generate the (half-)normal plot with a simulated envelope using the standard normal distribution (scale=F) or $N(\mu, \sigma^2)$ (scale=T).

Implementing a new model class is done by providing three functions to hnp: diagfun - to obtain model diagnostics, simfun - to simulate random variables and fitfun - to refit the model to simulated variables. The way these functions must be written is shown in the Examples section.

Value

hnp returns an object of class "hnp", which is a list containing the following components:

x	quantiles of the (half-)normal distribution
lower	lower envelope band
median	median envelope band
upper	upper envelope band
residuals	diagnostic measures in absolute value and in order
out.index	vector indicating which points are out of the envelope
col.paint.out	color of points which are outside of the envelope (used if paint.out=TRUE)
how.many.out	logical. Equals TRUE if how.many.out=TRUE in the hnp call
total	length of the diagnostic measure vector
out	number of points out of the envelope
print.on	logical. Equals TRUE if print.on=TRUE in the hnp call
paint.out	logical. Equals TRUE if paint.out=TRUE in the hnp call
all.sim	matrix with all diagnostics obtained in the simulations. Each column represents one simulation

Note

See documentation on example data sets for simple analyses and goodness-of-fit checking using hnp.

Author(s)

Rafael A. Moral <rafael_moral@yahoo.com.br>, John Hinde and Clarice G. B. Demétrio

References

Moral, R. A., Hinde, J. and Demétrio, C. G. B. (2017) Half-normal plots and overdispersed models in R: the hnp package. Journal of Statistical Software 81(10):1-23.

Atkinson, A. C. (1985) Plots, transformations and regression, Clarendon Press, Oxford.

Demétrio, C. G. B. and Hinde, J. (1997) Half-normal plots and overdispersion. GLIM Newsletter 27:19-26.

Hinde, J. and Demétrio, C. G. B. (1998) Overdispersion: models and estimation. Computational Statistics and Data Analysis 27:151-170.

Demétrio, C. G. B., Hinde, J. and Moral, R. A. (2014) Models for overdispersed data in entomology. In Godoy, W. A. C. and Ferreira, C. P. (Eds.) Ecological modelling applied to entomology. Springer.

See Also

plot.hnp, cbb, chryso, corn, fungi, oil, progeny, wolbachia

Examples

```
## Simple Poisson regression
set.seed(100)
counts <- c(rpois(5, 2), rpois(5, 4), rpois(5, 6), rpois(5, 8))
treatment <- gl(4, 5)
fit <- glm(counts ~ treatment, family=poisson)
anova(fit, test="Chisq")

## half-normal plot
hnp(fit)

## or save it in an object and then use the plot method
my.hnp <- hnp(fit, print.on=TRUE, plot=FALSE)
plot(my.hnp)

## changing graphical parameters
plot(my.hnp, lty=2, pch=4, cex=1.2)
plot(my.hnp, lty=c(2,3,2), pch=4, cex=1.2, col=c(2,2,2,1))
plot(my.hnp, main="Half-normal plot", xlab="Half-normal scores",
     ylab="Deviance residuals", legpos="bottomright")

## Using a numeric vector
my.vec <- rnorm(20, 4, 4)
hnp(my.vec) # using N(0,1)
```

```
hnp(my.vec, scale=TRUE) # using N(mu, sigma^2)

## Implementing new classes
## Users provide three functions - diagfun, simfun and fitfun,
## in the following way:
##
## diagfun <- function(obj) {
##   userfunction(obj, other_argumens)
##     # e.g., resid(obj, type="pearson")
##   }
##
## simfun <- function(n, obj) {
##   userfunction(n, other_arguments) # e.g., rpois(n, fitted(obj))
##   }
##
## fitfun <- function(y.) {
##  userfunction(y. ~ linear_predictor, other_arguments, data=data)
##    # e.g., glm(y. ~ block + factor1 * factor2, family=poisson,
##    #           data=mydata)
##  }
##
## when response is binary:
## fitfun <- function(y.) {
##  userfunction(cbind(y., m-y.) ~ linear_predictor,
##               other_arguments, data=data)
##    #e.g., glm(cbind(y., m-y.) ~ treatment - 1,
##    #          family=binomial, data=data)
##  }

## Not run:
## Example no. 1: Using Cook's distance as a diagnostic measure
y <- rpois(30, lambda=rep(c(.5, 1.5, 5), each=10))
tr <- gl(3, 10)
fit1 <- glm(y ~ tr, family=poisson)

# diagfun
d.fun <- function(obj) cooks.distance(obj)

# simfun
s.fun <- function(n, obj) {
  lam <- fitted(obj)
  rpois(n, lambda=lam)
}

# fitfun
my.data <- data.frame(y, tr)
f.fun <- function(y.) glm(y. ~ tr, family=poisson, data=my.data)

# hnp call
hnp(fit1, newclass=TRUE, diagfun=d.fun, simfun=s.fun, fitfun=f.fun)

## Example no. 2: Implementing gamma model using package gamlss
# load package
```

```
require(gamlss)

# model fitting
y <- rGA(30, mu=rep(c(.5, 1.5, 5), each=10), sigma=.5)
tr <- gl(3, 10)
fit2 <- gamlss(y ~ tr, family=GA)

# diagfun
d.fun <- function(obj) resid(obj) # this is the default if no
                                  # diagfun is provided

# simfun
s.fun <- function(n, obj) {
  mu <- obj$mu.fv
  sig <- obj$sigma.fv
  rGA(n, mu=mu, sigma=sig)
}

# fitfun
my.data <- data.frame(y, tr)
f.fun <- function(y.) gamlss(y. ~ tr, family=GA, data=my.data)

# hnp call
hnp(fit2, newclass=TRUE, diagfun=d.fun, simfun=s.fun,
    fitfun=f.fun, data=data.frame(y, tr))

## Example no. 3: Implementing binomial model in gamlss
# model fitting
y <- rBI(30, bd=50, mu=rep(c(.2, .5, .9), each=10))
m <- 50
tr <- gl(3, 10)
fit3 <- gamlss(cbind(y, m-y) ~ tr, family=BI)

# diagfun
d.fun <- function(obj) resid(obj)

# simfun
s.fun <- function(n, obj) {
  mu <- obj$mu.fv
  bd <- obj$bd
  rBI(n, bd=bd, mu=mu)
}

# fitfun
my.data <- data.frame(y, tr, m)
f.fun <- function(y.) gamlss(cbind(y., m-y.) ~ tr,
                             family=BI, data=my.data)

# hnp call
hnp(fit3, newclass=TRUE, diagfun=d.fun, simfun=s.fun, fitfun=f.fun)

## End(Not run)
```

hnp-internal *Internal function to prepare* hnp *objects*

Description

This is an internally called function used to prepare hnp objects for plotting

Author(s)

Rafael A. Moral <rafael_moral@yahoo.com.br>, John Hinde and Clarice G. B. Demétrio

newhnp *Method for non-implemented model classes*

Description

Uses user defined functions to produce the (half-)normal plot with simulated envelope.

Usage

```
newhnp(object, sim=99, conf=.95, halfnormal=T, plot.sim=T,
       verb.sim=F, how.many.out=F, print.on=F, paint.out=F,
       col.paint.out, diagfun, simfun, fitfun, ...)
```

Arguments

object	fitted model object or numeric vector.
sim	number of simulations used to compute envelope. Default is 99.
conf	confidence level of the simulated envelope. Default is 0.95.
halfnormal	logical. If TRUE, a half-normal plot is produced. If FALSE, a normal plot is produced. Default is TRUE.
plot.sim	logical. Should the (half-)normal plot be plotted? Default is TRUE.
verb.sim	logical. If TRUE, prints each step of the simulation procedure. Default is FALSE.
how.many.out	logical. If TRUE, the number of points out of the envelope is printed. Default is FALSE.
print.on	logical. If TRUE, the number of points out of the envelope is printed on the plot. Default is FALSE.
paint.out	logical. If TRUE, points out of the simulation envelope are plotted in a different color. Default is FALSE.
col.paint.out	If paint.out=TRUE, sets the color of points out of the envelope. Default is "red".
diagfun	user-defined function used to obtain the diagnostic measures from the fitted model object (only used when newclass=TRUE). Default is resid.

simfun	user-defined function used to simulate a random sample from the model estimated parameters (only used when newclass=TRUE).
fitfun	user-defined function used to re-fit the model to simulated data (only used when newclass=TRUE).
...	extra graphical arguments passed to plot.hnp.

Details

By providing three user-defined functions, newhnp produces the half-normal plot with simulated envelope for a model whose class is not yet implemented in the package.

The first function, diagfun, must extract the desired model diagnostics from a model fit object. The second function, simfun, must return the response variable (numeric vector or matrix), simulated using the same error distributions and estimated parameters from the fitted model. The third and final function, fitfun, must return a fitted model object. See the Examples section.

Author(s)

Rafael A. Moral <rafael_moral@yahoo.com.br>, John Hinde and Clarice G. B. Demétrio

Examples

```
## Implementing new classes
## Users provide three functions - diagfun, simfun and fitfun,
## in the following way:
##
## diagfun <- function(obj) {
##   userfunction(obj, other_argumens)
##     # e.g., resid(obj, type="pearson")
##   }
##
## simfun <- function(n, obj) {
##   userfunction(n, other_arguments) # e.g., rpois(n, fitted(obj))
##   }
##
## fitfun <- function(y.) {
##  userfunction(y. ~ linear_predictor, other_arguments, data=data)
##    # e.g., glm(y. ~ block + factor1 * factor2, family=poisson,
##    #           data=mydata)
##  }
##
## when response is binary:
## fitfun <- function(y.) {
##  userfunction(cbind(y., m-y.) ~ linear_predictor,
##               other_arguments, data=data)
##    #e.g., glm(cbind(y., m-y.) ~ treatment - 1,
##    #          family=binomial, data=data)
##  }

## Not run:
## Example no. 1: Using Cook's distance as a diagnostic measure
y <- rpois(30, lambda=rep(c(.5, 1.5, 5), each=10))
```

```
tr <- gl(3, 10)
fit1 <- glm(y ~ tr, family=poisson)

# diagfun
d.fun <- function(obj) cooks.distance(obj)

# simfun
s.fun <- function(n, obj) {
  lam <- fitted(obj)
  rpois(n, lambda=lam)
}

# fitfun
my.data <- data.frame(y, tr)
f.fun <- function(y.) glm(y. ~ tr, family=poisson, data=my.data)

# hnp call
hnp(fit1, newclass=TRUE, diagfun=d.fun, simfun=s.fun, fitfun=f.fun)

## Example no. 2: Implementing gamma model using package gamlss
# load package
require(gamlss)

# model fitting
y <- rGA(30, mu=rep(c(.5, 1.5, 5), each=10), sigma=.5)
tr <- gl(3, 10)
fit2 <- gamlss(y ~ tr, family=GA)

# diagfun
d.fun <- function(obj) resid(obj) # this is the default if no
                                  # diagfun is provided

# simfun
s.fun <- function(n, obj) {
  mu <- obj$mu.fv
  sig <- obj$sigma.fv
  rGA(n, mu=mu, sigma=sig)
}

# fitfun
my.data <- data.frame(y, tr)
f.fun <- function(y.) gamlss(y. ~ tr, family=GA, data=my.data)

# hnp call
hnp(fit2, newclass=TRUE, diagfun=d.fun, simfun=s.fun,
    fitfun=f.fun, data=data.frame(y, tr))

## Example no. 3: Implementing binomial model in gamlss
# model fitting
y <- rBI(30, bd=50, mu=rep(c(.2, .5, .9), each=10))
m <- 50
tr <- gl(3, 10)
fit3 <- gamlss(cbind(y, m-y) ~ tr, family=BI)
```

```
# diagfun
d.fun <- function(obj) resid(obj)

# simfun
s.fun <- function(n, obj) {
  mu <- obj$mu.fv
  bd <- obj$bd
  rBI(n, bd=bd, mu=mu)
}

# fitfun
my.data <- data.frame(y, tr, m)
f.fun <- function(y.) gamlss(cbind(y., m-y.) ~ tr,
                             family=BI, data=my.data)

# hnp call
hnp(fit3, newclass=TRUE, diagfun=d.fun, simfun=s.fun, fitfun=f.fun)

## End(Not run)
```

`oil` Diaphorina citri *oviposition data*

Description

Effects of three agricultural oils on *Diaphorina citri* oviposition.

Usage

```
data(oil)
```

Format

A data frame with 70 observations on the following 2 variables.

`y`	numeric	number of eggs laid
`treat`	factor	treatments applied in the experiment

Details

In an experiment to assess the effect of three agricultural oils on the oviposition of *Diaphorina citri*, seventy Orange Jessamine (*Murraya paniculata*) plants were sprayed with solutions of the mineral oils Oppa and Iharol, and the vegetable oil Nortox. The experiment used the oils in concentrations of 0.5 and 1.0 percent and a control of plain water set out in a completely randomized design with ten replicates. Following treatment, when the plants were dry, ten pregnant females of *D. citri* were released on each plant. After five days, the insects were removed and the total number of eggs on each plant was observed, see Amaral et al (2012). This is an example of aggregated data as the number of eggs is the sum over the (unrecorded) numbers of eggs deposited each day and the possibility of day to day variation may contribute additional variability to the recorded counts.

Source

Demétrio, C. G. B., Hinde, J. and Moral, R. A. (2014) Models for overdispersed data in entomology. In Godoy, W. A. C. and Ferreira, C. P. (Eds.) Ecological modelling applied to entomology. Springer.

References

Amaral, F. S. A., Poltronieri, A. S., Alves, E. B., Omoto, C. (2012) Efeito de oleos agricolas no comportamento de oviposicao e viabilidade de ovos de *Diaphorina citri Kuwayama* (*Hemiptera: Psyllidae*). In: XX Simposio Internacional de Iniciacao Cientifica da Universidade de Sao Paulo, 2012, Pirassununga

Examples

```
data(oil)

# Poisson fit
model1 <- glm(y ~ treat, family=poisson, data=oil)
anova(model1, test="Chisq")
sum(resid(model1, ty="pearson")^2)

# Quasi-Poisson fit
model2 <- glm(y ~ treat, family=quasipoisson, data=oil)
summary(model2)
anova(model2,test="F")
summary(model2)$dispersion

# Negative binomial fit
require(MASS)
model3 <- glm.nb(y ~ treat, data=oil)
thetahat <- summary(model3)$theta
anova(model3, test="F")

# half-normal plots
par(mfrow=c(1,3),cex=1.4, cex.main=0.9)
hnp(model1,pch=4, main="(a) Poisson",
     xlab="Half-normal scores", ylab="Deviance residuals")
hnp(model2,pch=4, main="(b) Quasi-Poisson",
     xlab="Half-normal scores", ylab='')
hnp(model3,pch=4, main="(c) Negative binomial",
     xlab="Half-normal scores", ylab='')

## Not run:
# using aods3
require(aods3)
model3b <- aodml(y ~ treat, family="nb", phi.scale="inverse",
                 fixpar=list(8, 1.086148), data=oil)
hnp(model3b)

## End(Not run)

## for discussion on the analysis of this data set,
## see Demetrio et al. (2014)
```

orange *Orange tissue-culture experiment data*

Description

Number of embryos of orange variety *Caipira* produced with different sugar types.

Usage

```
data("orange")
```

Format

A data frame with 150 observations on the following 4 variables.

`block` a factor with levels `1` `2` `3` `4` `5`

`sugar` a factor with levels `Maltose` `Glucose` `Lactose` `Galactose` `Sucrose` `Glycerol`

`dose` a numeric vector

`embryos` a numeric vector

Details

To study the effect of six sugars (maltose, glucose, galactose, lactose, sucrose and glycerol) on the stimulation of somatic embryos from callus cultures, the number of embryos after approximately four weeks was observed. The experiment was set up in a completely randomized block design with five blocks and the six sugars at dose levels of 18, 37, 75, 110 and 150 mM for the first five and 6, 12, 24, 36 and 50 mM for glycerol, see Tomaz (2001). The main interest was in the dose-response relationship.

Source

Tomaz ML, Mendes BMJ, Filho FAM, Demetrio CGB, Jansakul N, Rodriguez APM (1997). Somatic embryogenesis in *Citrus* spp.: Carbohydrate stimulation and histodifferentiation. In Vitro Cellular & Developmental Biology - Plant, 37, 446–452.

References

Moral, R. A., Hinde, J. and Demétrio, C. G. B. (2017) Half-normal plots and overdispersed models in R: the hnp package. Journal of Statistical Software 81(10):1-23.

Examples

```
data(orange)

require(gamlss)
fit_nbI <- gamlss(embryos ~ block + poly(dose, 2) * sugar,
                  family=NBII(), data=orange)
```

```
d.fun <- function(obj) resid(obj)

s.fun <- function(n, obj) {
  mu <- obj$mu.fv
  sigma <- obj$sigma.fv
  rNBII(n, mu, sigma)
}

f.fun <- function(y.) {
  gamlss(y. ~ block + poly(dose, 2) * sugar, family=NBII(), data=orange)
}

## Not run:
hnp(fit_nbI, newclass=TRUE, diagfun=d.fun, simfun=s.fun, fitfun=f.fun)

## End(Not run)

fit_pred <- gamlss(embryos ~ poly(dose, 2) * sugar, family=NBII(), data=orange)
orange.pred <- rbind(expand.grid(sugar=levels(orange$sugar)[-6], dose=18:150),
                     expand.grid(sugar="Glycerol", dose=6:50))
orange.pred$pred <- predict(fit_pred, newdata=orange.pred, type="response")
require(latticeExtra)
trellis.par.set(strip.background=list(col="lightgrey"))
xyplot(embryos ~ dose | sugar, scales=list(relation="free"), layout=c(3,2),
       data=orange, col=1, xlab="Dose levels", ylab="Number of embryos") +
  as.layer(xyplot(pred ~ dose | sugar, type="l", col=1, data=orange.pred))
```

`plot.hnp` *Plot Method for hnp Objects*

Description

The `plot` method for objects of class hnp.

Usage

```
## S3 method for class 'hnp'
plot(x, cex, pch, colour, lty, lwd, type,
     xlab, ylab, main, legpos, legcex, ...)
```

Arguments

`x`	object of class "hnp".
`cex`	character expansion size.
`pch`	character string or vector of one character or integer for plotting characters, see `points`.
`colour`	vector of colours.
`lty`	vector of line types.

lwd	vector of line widths.
type	type of plot for each envelope band and points. Default is c("l","l","l","p").
xlab	title for x axis, as in plot
ylab	title for y axis, as in plot
main	plot title.
legpos	if print.on=TRUE, represents the position where the information should be printed ("topright", "topleft", "bottomright", "bottomleft"), as in legend.
legcex	if print.on=TRUE, character expansion size of legend.
...	extra graphical arguments passed to matplot.

Value

None.

Author(s)

Rafael A. Moral <rafael_moral@yahoo.com.br>, John Hinde and Clarice G. B. Demétrio

References

Moral, R. A., Hinde, J. and Demétrio, C. G. B. (2017) Half-normal plots and overdispersed models in R: the hnp package. Journal of Statistical Software 81(10):1-23.

Demétrio, C. G. B. and Hinde, J. (1997) Half-normal plots and overdispersion. GLIM Newsletter 27:19-26.

Hinde, J. and Demétrio, C. G. B. (1998) Overdispersion: models and estimation. Computational Statistics and Data Analysis 27:151-170.

Demétrio, C. G. B., Hinde, J. and Moral, R. A. (2014) Models for overdispersed data in entomology. In Godoy, W. A. C. and Ferreira, C. P. (Eds.) Ecological modelling applied to entomology. Springer.

See Also

hnp

Examples

```
## Simple Poisson regression
set.seed(100)
counts <- c(rpois(5, 2), rpois(5, 4), rpois(5, 6), rpois(5, 8))
treatment <- gl(4, 5)
fit <- glm(counts ~ treatment, family=poisson)
anova(fit, test="Chisq")

## half-normal plot
hnp(fit)

## or save it in an object and then use the plot method
my.hnp <- hnp(fit, print.on=TRUE, plot=FALSE)
```

```
plot(my.hnp)

## changing graphical parameters
plot(my.hnp, lty=2, pch=4, cex=1.2)
plot(my.hnp, lty=c(2,3,2), pch=4, cex=1.2, col=c(2,2,2,1))
plot(my.hnp, main="Half-normal plot", xlab="Half-normal scores",
     ylab="Deviance residuals", legpos="bottomright")
```

`progeny` Sitophilus zeamais *progeny*

Description

Progeny of *Sitophilus zeamais*, the maize weevil, when treated with different organic extracts

Usage

```
data(progeny)
```

Format

A data frame with 40 observations on the following 2 variables.

`extract`	factor	levels `leaf`, `branch`, `seed` and `control`
`y`	numeric	number of emerged insects after 60 days

Details

Petri dishes containing 10g of corn were treated with extracts prepared with different parts of the plant *Annona mucosa* (seeds, leaves and branches) at a concentration of 1500 mg/kg or just water (control), using a completely randomized design with 10 replicates. Then 20 *S. zeamais* adults were placed in each Petri dish and the focus is on the numbers of emerged insects (progeny) after 60 days, see Ribeiro et al (2013).

Source

Demétrio, C. G. B., Hinde, J. and Moral, R. A. (2014) Models for overdispersed data in entomology. In Godoy, W. A. C. and Ferreira, C. P. (Eds.) Ecological modelling applied to entomology. Springer.

References

Ribeiro, L. P., Vendramin, J. D., Bicalho, K. U., Andrade, M. S., Fernandes, J. B., Moral, R. A., Demétrio, C. G. B. (2013) Annona mucosa Jacq. (Annonaceae): A promising source of bioactive compounds against Sitophilus zeamais Mots. (Coleoptera: Curculionidae). J Stored Prod Res 55:6-14

Moral, R. A., Hinde, J. and Demétrio, C. G. B. (2017) Half-normal plots and overdispersed models in R: the hnp package. Journal of Statistical Software 81(10):1-23.

Examples

```
data(progeny)

# Poisson fit
model1 <- glm(y ~ extract, family=poisson, data=progeny)
anova(model1, test="Chisq")

# Quasi-Poisson fit
model2 <- glm(y ~ extract, family=quasipoisson, data=progeny)
summary(model2)$dispersion
anova(model2, test="F")

# half-normal plots
par(mfrow=c(1,2),cex=1.4, cex.main=0.9, pty='s')
hnp(model1, pch=4, main="(a) Poisson; log-linear",
    xlab="Half-normal scores", ylab="Deviance residuals")
hnp(model2, pch=4, main="(b) Quasi-Poisson; log-linear",
    xlab="Half-normal scores", ylab="Deviance residuals")

anova(model1, test="Chisq") # Poisson model
anova(model2, test="F") # quasi-Poisson model
summary(model1) # Poisson model
summary(model2) # quasi-Poisson model

# now with factor level parameterisation
summary(update(model1,.~.-1))
summary(update(model2,.~.-1))

## for discussion on the analysis of this data set,
## see Demetrio et al. (2014)
```

`wolbachia` Trichogramma galloi *parasitism data*

Description

Viability of *Trichogramma galloi*, a parasitoid wasp, when infected with *Wolbachia*, a bacteria known to change its reproductive aspects.

Usage

```
data(wolbachia)
```

Format

A data frame with 106 observations on the following 3 variables.

`y`	numeric; number of eggs with an orifice
`m`	numeric; total number of parasitised eggs

`treat` a factor with levels `m+f+`, `m+f-`, `f+`, `m-f-` and `f-`,
where `m` stands for male, `f` stands for female, `+` means infected and `-` means non-infected;
`f+` and `f-` represent virgin infected and non-infected females, respectively.

Details

The bacteria *Wolbachia* is commonly found in various insect species and has the ability to change reproductive aspects of its host. When it infects the wasp *Trichogramma galloi* it is known to induce thelythokous parthenogenesis, i.e., only females are produced from unfertilized eggs. Souza (2011) conducted an experiment to assess the effects of *Wolbachia* on the viability of *T. galloi* eggs. Around 100 *Diatraea saccharalis* eggs (the host) were offered to infected (+) or non-infected (-) parasitoid couples or virgin females every day until the death of the female. The parasitised eggs, easily identifiable because they become dark, were then kept on moist filter paper for twenty days when counts were then made of the number of eggs that had an orifice, which meant that an adult parasitoid had emerged and thus the parasitoid was viable.

Source

Demétrio, C. G. B., Hinde, J. and Moral, R. A. (2014) Models for overdispersed data in entomology. In Godoy, W. A. C. and Ferreira, C. P. (Eds.) Ecological modelling applied to entomology. Springer.

References

Souza, A. R. (2011) A interacao *Wolbachia - Trichogramma galloi Zucchi*, 1988 (*Hymenoptera: Trichogrammatidae*). Master's dissertation, ESALQ-USP

Examples

```
data(wolbachia)

# Binomial fit
model1 <- glm(cbind(y, m-y) ~ treat, family=binomial, data=wolbachia)
anova(model1, test="Chisq")

# Quasi-binomial fit
model2 <- glm(cbind(y, m-y) ~ treat, family=quasibinomial, data=wolbachia)
summary(model2)
anova(model2,test="F")

## half normal plots
par(mfrow=c(1,2),cex=1.2, cex.main=1.1)
hnp(model1, print=TRUE, pch=4, main="(a) Binomial",
    xlab="Half-normal scores", ylab="Deviance residuals")
hnp(model2, print=TRUE, pch=4, main="(b) Quasi-binomial",
    xlab="Half-normal scores", ylab='')

## Not run:
# Beta-binomial fit
### using package aods3
require(aods3)
```

```
model3 <- aodml(cbind(y, m-y) ~ treat, family='bb', data=wolbachia)
hnp(model3, print=TRUE, pch=4,
    xlab="Half-normal scores", ylab='Deviance residuals')

### using package VGAM
require(VGAM)

model3a <- vglm(cbind(y, m-y) ~ treat, family=betabinomial,
                data=wolbachia)
summary(model3a)
1/(1+exp(-coef(model3a)[2])) # phi estimate
hnp(model3a, data=wolbachia)

## End(Not run)

## for discussion on the analysis of this data set,
## see Demetrio et al. (2014)
```

Index