Package ‘ecostats’

August 24, 2022

Title Code and Data Accompanying the Eco-Stats Text (Warton 2022)

Version 1.1.11

Date 2022-08-23

Description Functions and data supporting the Eco-Stats text (Warton, 2022, Springer), and solutions to exercises. Functions include tools for using simulation envelopes in diagnostic plots, and a function for diagnostic plots of multivariate linear models. Datasets mentioned in the package are included here (where not available elsewhere) and there is a vignette for each chapter of the text with solutions to exercises.

License LGPL (>= 2.1)

Imports ecoCopula, GET, graphics, grDevices, MASS, mgcv, mvtnorm, stats

Encoding UTF-8

LazyData true

RoxygenNote 7.2.1

Depends R (>= 3.5.0), mvabund (>= 4.2)

Suggests knitr, rmarkdown, ade4, caper, car, corrplot, DAAG, DHARMa, dplyr, gclus, GGally, ggplot2, ggthemes, glmnet, gllvm, glmmTMB, GPArotation, grplasso, lattice, leaps, lme4, MCMCglmm, multcomp, nlme, ordinal, permute, pgirmess, phylobase, phylosignal, psych, reshape2, smatr, testthat, vegan, VGAM, covr

VignetteBuilder knitr

NeedsCompilation no

Author David Warton [aut, cre],
Christopher Chung [ctb],
Mark Donoghoe [ctb],
Eve Slavich [ctb]

Maintainer David Warton <david.warton@unsw.edu.au>

Repository CRAN

Date/Publication 2022-08-24 06:50:02 UTC

R topics documented:

`addSmooth` *addSmooth Add a smoother to a plot, with pointwise confidence band*

Description

Adds a smoother to a plot of `y` against `x`, and a confidence band around the smoother with pointwise coverage (**not** global, unlike `plotenvelope`). The smoother is constructed using `gam` and confidence bands use Wald intervals, extracting the standard error from `predict.gam`.

Usage

```
addSmooth(
  x,
  y,
  conf.level = 0.05,
  line.col = "slateblue4",
  envelope.col = adjustcolor(line.col, alpha.f = 0.1),
  ...
)
```

Arguments

`x`	is the `x` co-ordinates for the plot, *e.g.* the fitted values in a plot of residuals vs fitted values.
`y`	is the `y` co-ordinates for the plot
`conf.level`	the confidence level to use in constructing the confidence band around the smoother.
`line.col`	color of smoother. Defaults to "slateblue4".
`envelope.col`	color of the global envelope around the expected trend. All data points should always stay within this envelope (and will for a proportion `conf.level` of datasets satisfying model assumptions). If a smoother has been used on the residual vs fits or scale-location plot, the envelope is constructed around the smoother, that is, the smoother should always stay within the envelope.
`...`	further arguments sent through to `lines` and `polygon`.

Details

A challenge when interpreting diagnostic plots is understanding the extent to which deviations from the expected pattern could be due to random noise (sampling variation) rather than actual assumption violations. This function is intended to assess this, by simulating multiple realizations of residuals (and fitted values) in situations where assumptions are satisfied, and plotting a global simulation envelope around these at level `conf.level`.

This function adds a smoother to a plot of `y` against `x` using `gam`, and a confidence band around the smoother with pointwise coverage (**not** global, unlike `plotenvelope`) using Wald intervals that take the standard error of predicted values from `predict.gam`.

When constructing a plot to diagnose model fit, `plotenvelope` is preferred, because this constructs simulation envelopes globally, simplifying interpretation, and tends to give more accurate envelopes, via simulation. This function is intended to fit trends to data for descriptive purposes, but it could be used (with caution) to diagnose model fit in residual plots in situations where `plotenvelope` is unavailable.

Value

A smoother is added to the existing plot, with confidence band. In addition, a data frame is invisibly returned with the following components:

`X` The values of `x` at which the smoother is evaluated.

`Yhat` Values of the smoother used to predict `y`, for each value of `x`.

`Yhi` The upper bound on the confidence band, for each value of `x`.

`Ylo` The lower bound on the global envelope, for each value of `x`.

Author(s)

David Warton <david.warton@unsw.edu.au>

References

Warton DI (2022) Eco-Stats - Data Analysis in Ecology, from *t*-tests to multivariate abundances. Springer, ISBN 978-3-030-88442-0

See Also

plotenvelope

Examples

```
data(globalPlants)
plot(log(height)~lat,data=globalPlants)
with(globalPlants, addSmooth(lat,log(height)) )
```

---

anovaPB *Parametric bootstrap testing to compare two models by analysis of variance (deviance)*

Description

Compute analysis of variance (or deviance) tables for two fitted model objects, with the *P*-value estimated using a parametric bootstrap – by repeatedly simulating new responses from the fitted model under the null hypothesis.

Usage

```
anovaPB(
  objectNull,
  object,
  n.sim = 999,
  colRef = switch(class(object)[1], lm = 5, lmerMod = 6, 4),
  rowRef = 2,
  ...
)
```

Arguments

objectNull	is the fitted model under the null hypothesis. This can be *any* object that responds to simulate, update and anova.
object	is the fitted model under the alternative hypothesis. This can be *any* object that responds to update, anova and model.frame. It must be larger than objectNull and its model frame should contain all the terms required to fit objectNull.
n.sim	the number of simulated datasets to generate for parametric bootstrap testing. Defaults to 999.
colRef	the column number where the test statistic of interest can be found in the standard anova output when calling anova(objectNull,object). Default choices have been set for common models (colRef=5 for lm objects, colRef=6 for lmer objects and colRef=4 otherwise, which is correct for glm and gam objects).
rowRef	the row number where the test statistic of interest can be found in the standard anova output when calling anova(objectNull,object). Defaults to 2, because it is on the second row for most common models.
...	further arguments sent through to anova.

Details

The anova function typically relies on classical asymptotic results which sometimes don't apply well, e.g. when using penalised likelihood to fit a generalised additive model, or when testing whether a random effect should be included in a mixed model. In such cases we can get a more accurate test by using simulation to estimate the *P*-value – repeatedly simulating new sets of simulated responses, refitting the null and alternative models, and recomputing the test statistic. This process allows us to estimate the distribution of the test statistic when the null hypothesis is true, hence draw a conclusion about whether the observed test statistic is large relative to what would be expected under the null hypothesis. The process is often referred to as a *parametric bootstrap* (Davison & Hinkley 1997), which the *PB* in the function name (anovaPB) is referring to.

This function will take any pair of fitted models, a null (objectNull) and an alternative (object), and use simulation to estimate the *P*-value of the test of whether there is evidence against the model in objectNull in favour of the model in object. This function works by repeatedly performing an anova to compare objectNull to object, where the responses in the model.frame have been replaced with values simulated by applying simulate to objectNull. Hence for this function to work, the objects being compared need to respond to the anova, simulate and model.frame functions.

This function needs to be able to find the test statistic in the anova output, but it is stored in different places for different types of objects. It is assumed that anova(objectNull,object) returns a matrix and that the test statistic is stored in the (rowRef,colRef)th element, where both rowRef and colRef have been chosen to be correct for common model types (glm, gam, lmer).

Value

A matrix which has the appearance and behaviour of an object returned by anova(objectNull,object), except that the *P*-value has been computed by parametric bootstrap, and the matrix now inherits class anovaPB.

Author(s)

David Warton <david.warton@unsw.edu.au>

References

Davison A.C. & Hinkley D. V. (1997) Bootstrap methods and their application, Cambridge University Press. Warton DI (2022) Eco-Stats - Data Analysis in Ecology, from *t*-tests to multivariate abundances. Springer, ISBN 978-3-030-88442-0

See Also

anova

Examples

```
# fit a Poisson regression to random data:
y = rpois(50,lambda=1)
x = 1:50
rpois_glm = glm(y~x,family=poisson())
rpois_int = glm(y~1,family=poisson())
```

```
anovaPB(rpois_int,rpois_glm,n.sim=99)
```

aphids *Aphid data*

Description

A study of the effects of bird exclusion on biological control of aphids in oat and wheat fields in Germany (Grass *et al.* 2017). Many thanks to Ingo for providing the raw data. In each of two fields (one of oats, one of wheat) there were eight plots, four with plastic netting to exclude birds, and four without. Aphid abundance was counted on seven different occasions over the first 38 weeks following netting. The expectation was that aphid numbers would decrease on bird exclusion, because an important food source to tree sparrows is aphid predators, hoverflies and ladybird beetles, so presence of birds may be limit the effectiveness of a biological control of aphids.

Usage

```
data(aphids)
```

Format

A list containing two dataframes, `oat` and `wheat`, depending on the crop. Each dataframe contains:

Plot The plot ID, a factor with eight levels

Treatment A factor indicating whether birds were excluded, `excluded` or `present`.

Time The number of days since netting was applied.

counts Aphid abundance (counts)

logcount log(y+1)-transformed aphid abundance

References

Grass *et al.* (2017) Insectivorous birds disrupt biological control of cereal aphids. Ecology, **98** 1583-90.

Examples

```
data(aphids)
cols=c(rgb(1,0,0,alpha=0.5),rgb(0,0,1,alpha=0.5)) #transparent colours
with(aphids$oat, interaction.plot(Time,Plot,logcount,legend=FALSE,
                              col=cols[Treatment], lty=1, ylab="Counts [log(y+1) scale]",
                                xlab="Time (days since treatment)") )
                             legend("bottomleft",c("Excluded","Present"),col=cols,lty=1)
```

`aphidsBACI` *Aphid data as a BACI design*

Description

A subset from a study of the effects of bird exclusion on biological control of aphids in a German oat field (Grass *et al.* 2017). Many thanks to Ingo for providing the raw data. The full data are in `aphids`, here we provide just two sampling times, 3 days and 30 days after netting. OK, this is not quite a BACI design, 3 days after netting is not quite "before" but effects at this point should be negligible...

There were eight plots, four with plastic netting to exclude birds, and four without. Aphid abundance was counted on 5 shoots per plot. The expectation was that aphid numbers would decrease on bird exclusion, because an important food source to tree sparrows is aphid predators, hoverflies and ladybird beetles, so presence of birds may be limit the effectiveness of a biological control of aphids.

Usage

```
data(aphidsBACI)
```

Format

A dataframe containing:

Plot The plot ID, a factor with eight levels

Treatment A factor indicating whether birds were excluded, `excluded` or `present`.

Time The data of sampling (June 18th or July 15th, 3 and 30 days after netting, respectively)

counts Aphid abundance (counts)

logcount log(y+1)-transformed aphid abundance

References

Grass *et al.* (2017) Insectivorous birds disrupt biological control of cereal aphids. Ecology, **98** 1583-90.

Examples

```
data(aphidsBACI)
plot(logcount~interaction(Treatment,Time),data=aphidsBACI,col=c("lightgreen","pink"))
```

BlockBootID *Construct a moving block bootstrap resampling scheme*

Description

A spatial moving block bootstrap resampling scheme is constructed, that can then be used in resampling-based inference (e.g. using `anova.manyglm`). Blocks are constructed as discs or tiles of observations of a user-specified size and shape.

Usage

```
BlockBootID(
  x,
  y,
  block_L,
  nBoot = 499,
  Grid_space,
  lookuptables.folderpath = NA,
  shape = "disc",
  ...
)
```

Arguments

`x`	the easting of spatial coordinate for the site.
`y`	the northing of the spatial coordinate for the site.
`block_L`	the size of the spatial blocks to be resampled.
`nBoot`	the number of resamples required (defaults to 499).
`Grid_space`	the resolution of the lattice used to sample centres of spatial blocks. Defaults to a third of the resolution of `block_Ls`.
`lookuptables.folderpath`	(optional) path to a directory where lookup tables can be found that assign observations to spatial blocks. Such tables would considerably speed up the process.
`shape`	shape of the spatial blocks to resample. Default is `disc`, but `square` tiles can also be specified.
`...`	additional arguments passed to `resample_blocks_by_area`.

Details

A spatial moving block bootstrap resampling scheme is constructed, that can then be used in resampling-based inference. A matrix of IDs is returned, with different resamples in different rows of the matrix, which can be used as input to functions designed for resampling-based inference (such as `anova.manyglm`).

Blocks are constructed as discs or tiles of observations, whose size is controlled by `blockl_L`, these blocks are kept together in resampling. The centre of each block is chosen at random along a lattice specified via `Grid_space`.

The most computationally intensive part of this process is working out which observations belong in which blocks. If repeated analyses are to be undertaken in the same spatial domain using the same sites, it is best to run this process only once and save the result as a set of lookup tables. Then the path to the directory containing these tables is specified via `lookuptables.folderpath` and the whole thing runs heaps faster.

Value

A matrix of IDs for each moving block bootstrap resample, with different resamples in rows and observations in columns.

Author(s)

Eve Slavich <eve.slavich@unsw.edu.au>

See Also

`anova.manyglm`, `lm`

Examples

```
data(Myrtaceae)
# fit a lm:
library(mvabund)
Myrtaceae$logrich=log(Myrtaceae$richness+1)
mft_richAdd = manylm(logrich~soil+poly(TMP_MAX,degree=2)+
                     poly(TMP_MIN,degree=2)+poly(RAIN_ANN,degree=2),
                                       data=Myrtaceae)

# construct a boot ID matrix:
BootID = BlockBootID(x = Myrtaceae$X, y = Myrtaceae$Y, block_L = 20,
            nBoot = 199, Grid_space = 5)
anova(mft_richAdd,resamp="case",bootID=BootID)
```

`cpredict` *Conditional Predictions for Multivariate Linear Model Fits*

Description

Predicted values using full conditional models derived from a multivariate linear model (`mlm`) object. The full conditionals model each response as a linear model with all other responses used as predictors in addition to the regressors specified in the formula of the `mlm` object.

Usage

```
cpredict(object, standardize = TRUE, ...)
```

Arguments

`object`	a `mlm` object, typically the result of calling `lm` with a matrix response.
`standardize`	logical defaults to `TRUE`, standardising responses so they are comparable across responses.
`...`	further arguments passed to `predict.lm`, in particular, `newdata`. However, this function was not written to accept non-default values for `se.fit`, `interval` or `terms`.

Details

Predictions using an `mlm` object but based on the full conditional model, that is, from a linear model for each response as a function of all responses as well as predictors. This can be used in plots to diagnose the multivariate normality assumption.

By default predictions are standardised to facilitate overlay plots of multiple responses, as in `plotenvelope`.

This function behaves much like `predict.lm`, but currently, standard errors and confidence intervals around predictions are not available.

Value

A matrix of predicted values from full conditional models.

Author(s)

David Warton <david.warton@unsw.edu.au>

References

Warton DI (2022) Eco-Stats - Data Analysis in Ecology, from *t*-tests to multivariate abundances. Springer, ISBN 978-3-030-88442-0

See Also

`cresiduals`, `lm`, `plotenvelope`, `predict.lm`

Examples

```
data(iris)
# fit a mlm:
iris.mlm=lm(cbind(Sepal.Length,Sepal.Width,Petal.Length,Petal.Width)~Species,data=iris)
# predict each response conditionally on the values of all other responses:
cpredict(iris.mlm)
```

`cresiduals` *Extract Conditional Residuals from Multivariate Linear Model Fits*

Description

Residuals from full conditionals of a Multivariate Linear Model (`mlm`) object. The full conditional for each response is a linear model with all other responses used as predictors in addition to the regressors specified in the formula of the `mlm` object. This is used to diagnose the multivariate normality assumption in `plotenvelope`.

Usage

```
cresiduals(object, standardize = TRUE, ...)
```

Arguments

`object`	a `mlm` object, typically the result of calling `lm` with a matrix response.
`standardize`	logical defaults to `TRUE`, to return studentized residuals using `rstandard` so they are comparable across responses.
`...`	further arguments passed to `residuals.lm` or `rstandard`.

Details

A `residuals` function for `mlm` objects, which returns residuals from a full conditional model, that is, a linear model of each response against all responses as well as predictors, which can be used to diagnose the multivariate normality assumption. These can be standardized (`standardize=TRUE`) to facilitate overlay plots of multiple responses, as in `plotenvelope`.

Value

A matrix of residuals

Author(s)

David Warton <david.warton@unsw.edu.au>

References

Warton DI (2022) Eco-Stats - Data Analysis in Ecology, from *t*-tests to multivariate abundances. Springer, ISBN 978-3-030-88442-0

See Also

`cpredict`, `lm`, `plotenvelope`, `residuals`, `rstandard`

Examples

```
data(iris)
# fit a mlm:
iris.mlm=lm(cbind(Sepal.Length,Sepal.Width,Petal.Length,Petal.Width)~Species,data=iris)
# construct full conditional residuals:
cresiduals(iris.mlm)
```

`estuaries` *Effect of pollution on marine microinvertebrates in estuaries*

Description

Data from an observational study of whether there is a different in microinvertebrate communities between estuaries that have been heavily modified by human activity and those that have not, across seven estuaries along the coast of New South Wales, Australia (Clark et al. 2015).

Usage

```
data(estuaries)
```

Format

A dataframe containing (amongst other things):

Mod A factor describing whether the sample was taken from a 'Modified' or 'Pristine' estuary.

Zone Whether the sample was taken from Inner (upstream) or Outer (downstream) zone of the estuary.

Estuary A factor with seven levels identifying which estuary the sample was taken from.

Total Total abundance of all invertebrates in the sample

Richness Richness of taxa in the sample – the number of responses (of those in columns 8-94) taking a non-zero value

Other variables in the dataset give invertebrate counts separately for different taxa.

References

Clark, G. F., Kelaher, B. P., Dafforn, K. A., Coleman, M. A., Knott, N. A., Marzinelli, E. M., & Johnston, E. L. (2015). What does impacted look like? high diversity and abundance of epibiota in modified estuaries. Environmental Pollution 196, 12-20.

Examples

```
data(estuaries)
plot(Total~Estuary,data=estuaries,col=c(4,2,2,4,2,4,2))
```

`estuaryZone` *Effect of pollution on marine microinvertebrates in estuaries in different zones*

Description

Data from an observational study of whether there is a different in microinvertebrate communities between estuaries that have been heavily modified by human activity and those that have not, across seven estuaries along the coast of New South Wales, Australia (Clark et al. 2015). Sampling was undertaken in both Inner and Outer zones of the estuary.

Usage

```
data(estuaryZone)
```

Format

A dataframe containing (amongst other things):

Mod A factor describing whether the sample was taken from a 'Modified' or 'Pristine' estuary.

Zone Whether the sample was taken from Inner (upstream) or Outer (downstream) zone of the estuary.

Estuary A factor with seven levels identifying which estuary the sample was taken from.

Total Total abundance of all invertebrates in the sample

Richness Richness of taxa in the sample – the number of responses (of those in columns 8-94) taking a non-zero value

Other variables in the dataset give invertebrate counts separately for different taxa.

References

Clark, G. F., Kelaher, B. P., Dafforn, K. A., Coleman, M. A., Knott, N. A., Marzinelli, E. M., & Johnston, E. L. (2015). What does impacted look like? high diversity and abundance of epibiota in modified estuaries. Environmental Pollution **196**, 12-20.

Examples

```
data(estuaryZone)
cols=c("blue","red","lightblue","pink")
plot(Total~interaction(Estuary,Zone),data=estuaryZone,col=cols[c(1,2,2,1,2,1,2,3,4,4,3,4,3,4)])
```

`globalPlants` *Global Plants data*

Description

Data inspired by a worldwide study of plant height and its association with climate and latitude (Moles *et al.* 2009). Height is recorded for one dominant plant at each of 178 sites, along with dozens of climatic variables extracted from BIOCLIM. Sites, climate data and species lists come from Moles *et al.* (2009) but height data are not actually field measurements, due to data unavailability. Instead height has been taken from species descriptions available on Google, accessed 7/1/21.

Usage

```
data(globalPlants)
```

Format

A dataframe containing a whole bunch of stuff, the main things being:

height Maximum plant height, in metres, taken from species descriptions on Google (where available)

lat latitude of the site this plant was found at

temp average daily maximum temperature

rain total annual precipitation

rain.wetm Rainfall in the wettest month

References

Moles *et al.* (2009) Global patterns in plant height. Journal of Ecology **97**, 923-932.

Examples

```
data(globalPlants)
plot(height~lat,data=globalPlants)
```

`guineapig` *Guineapig data*

Description

Data derived from a study of the effects of nicotine on the development of guineapig offspring (Johns et al 1993). Ten pregnant guineapigs were injected with nicotine hydrogen tartite solution, and ten control guinea pigs were injected with saline solution as a control. Learning capabilities of offspring were then measured as the number of errors made when trying to remember where to find food in a simple maze. Data presented are not from the original paper- they have been generated to match the summary statistics reported in Johns et al (1993).

Usage

```
data(guineapig)
```

Format

A list containing two dataframes, `oat` and `wheat`, depending on the crop. Each dataframe contains:

errors Number of errors made

treatment N=Nicotine, C=Control

References

Johns *et al.* (1993) The effects of chronic prenatal exposure to nicotine on the behavior of guinea pigs (*Cavia porcellus*). The Journal of General Psychology **120**, 49-63.

Examples

```
data(guineapig)
plot(errors~treatment,data=guineapig)
```

`headbobLizards` *Headbob displays of* Anolis *lizards*

Description

Displays of 14 male *Anolis* lizards were recorded in Puerto Rican forest (Ord *et al. 2016*), along with key environmental characteristics. These lizards bob their head up and down (and do push-ups) in attempts to attract the attention of females, and advertise territory ownership to other males. How fast a lizard bobs its head was recorded, and it was of interest to understand which environmental features (out of temperature, light and noisiness) were related to head bobbing speed.

Usage

```
data(headbobLizards)
```

Format

A dataframe containing:

LizardID A factor indicating Which lizard we are talking about. There is one observation per lizard (but more were originally collected).

TemperatureC Temperature, in degrees Celsius, at the location where lizard was first observed.

AmbientLight Ambient light recorded using a handheld sensor pointed towards the bobbing is happening.

Bg_noise_max Visual background noise measured using video analysis.

Hbspd_max Maximum head-bobbing speed.

time Date and time of observation.

References

Ord, T. J., Charles, G. K., Palmer, M. & Stamps, J. A. (2016). Plasticity in social communication and its implications for the colonization of novel habitats. Behavioral Ecology **27**, 341-351

Examples

```
data(headbobLizards)
plot(Hbspd_max~Bg_noise_max, data=headbobLizards)
```

`maunaloa` *Atmospheric carbon dioxide concentration from the Mauna Loa Observatory*

Description

Monthly average measurements of carbon dioxide concentration from the Mauna Loa Observatory in Hawaii, from March 1958 to February 2021. Data available courtesy of the Global Monitoring Laboratory at the National Oceanic and Atmospheric Administration (NOAA) in the United States (https://www.esrl.noaa.gov/gmd/ccgg/trends/data.html).

Usage

```
data(maunaloa)
```

Format

A dataframe containing:

Date The data of the measurement in date format. One measurement is available for each month, the first day of the month is assumed here.

year The year of the measurement.

month The month of the measurement.

DateNum The date in numerical format, as `year+month/12`.

co2 Carbon dioxide measurement in parts per million. Calculated as the average of all daily measurements for the month.

Examples

```
data(maunaloa)
plot(co2~Date, type="l", data=maunaloa)
```

`Myrtaceae` *Species richness of* Myrtaceae *plants*

Description

Data derived from NSW National Parks and Wildlife Service resources on species richness for members of the *Myrtaceae* family (eucalypts and relates species) in 1000 monitoring transects west of Sydney in the Greater Blue Mountains World Heritage Area. Also included is soil type classified into 9 categories, and a few climate variables derived from Worldclim.

Usage

```
data(Myrtaceae)
```

Format

A dataframe containing (amongst other things):

X Easting of the site (in km).

Y Nothing of the site (in km).

richness Total number of *Mrtaceae* species observed at this site.

TMP_MAX Annual average of daily maximum temperature (degrees Celsius).

TMP_MIN Annual average of daily minimum temperature (degrees Celsius).

RAIN_ANN Annual precipitation (in millimetres).

soil Soil type, classified into nine categories.

aspect Aspect of the site (in degrees, 0=360=due North).

Examples

```
data(Myrtaceae)
library(ggplot2)
ggplot(Myrtaceae, aes(x=X, y=Y))+geom_point(aes(color=richness))+
  theme_classic()+xlab("West<-->East (km)")+ylab("South<-->North (km)")+
  scale_color_gradient(low="lightgreen",high="black")+
  labs(color="Species richness [log(y+1)-scale]")+theme(legend.position="top")+
  guides(color = guide_colorbar(title.position = "top",ticks=FALSE,
                             barwidth=15,barheight=0.5))+coord_fixed()
```

plotenvelope *Diagnostic Plots for a Fitted Object with Simulation Envelopes*

Description

Produces diagnostic plots of a fitted model y, and adds global envelopes constructed by simulating new residuals, to see how departures from expected trends compare to what might be expected if the fitted model were correct. Global envelopes are constructed using the GET package (Myllymäki et al 2017) for simultaneous control of error rates over the whole plot, by simulating new responses from the fitted model then recomputing residuals (which can be computationally intensive), or alternatively, by simulating residuals directly from the (multivariate) normal distribution (faster, but not always a smart move). Options for diagnostic plots to construct are a residual vs fits, a normal quantile plot, or scale-location plot, along the lines of plot.lm.

Usage

```
plotenvelope(
  y,
  which = 1:2,
  sim.method = "refit",
  n.sim = 199,
  conf.level = 0.95,
  type = "st",
  overlay = TRUE,
  transform = NULL,
 main = c("Residuals vs Fitted Values", "Normal Quantile Plot", "Scale-Location Plot"),
  xlab = c("Fitted values", "Theoretical Quantiles", "Fitted Values"),
  ylab = c("Residuals", "Residuals", expression(sqrt("|Residuals|"))),
  col = NULL,
  line.col = if (add.smooth) c("slateblue4", "olivedrab", "slateblue4") else
    rep("olivedrab", 3),
  envelope.col = adjustcolor(line.col, 0.1),
  add.smooth = TRUE,
  plot.it = TRUE,
  resFunction = NULL,
  predFunction = NULL,
  fitMin = if (inherits(y, "glm") | inherits(y, "manyglm")) -6 else -Inf,
  ...
)
```

Arguments

y	is *any* object that responds to residuals, predict and (if sim.method="refit") simulate and update.
which	a vector specifying the diagnostic plots to construct: 1. residual vs fits, with a smoother

2. normal quantile plot
3. scale-location plot, with smoother

These are the first three options in `plot.lm` and a little is borrowed from that code. A global envelope is included with each plot around where we expect to see the data (or the smoother, if requested, for plots 1 and 3) when model assumptions are satisfied. If not fully captured by the global envelope, there is some evidence that the model assumptions are not satisfied.

`sim.method`	How should residuals be simulated? The default for most objects is `"refit"`, which constructs new responses (via `simulate`), refits the model (via `update`), then recomputes residuals, often known as a *parametric bootstrap*. This is computationally intensive but gives a robust answer. This is the only suitable option if residuals are not expected to be normal when assumptions are satisfied (like when using `glm` with a non-Gaussian family). Alternatively, `"norm"` simulates from a normal distribution, matches means and variances (and covariances for multivariate responses) to the observed residuals. The `"stand.norm"` option sets means to zero and variances to one, which is appropriate when residuals should be standard normal when assumptions are satisfied (as for any object fitted using the `mvabund` package, for example). These options are faster but more approximate than `"refit"`, in fact `"stand.norm"` is used as the default for `manyglm` objects, for computational reasons.
`n.sim`	the number of simulated sets of residuals to be generated, to which the observed residuals will be compared. The default is 199 datasets.
`conf.level`	the confidence level to use in constructing the envelope.
`type`	the type of global envelope to construct, see `global_envelope_test` for details. Default `"st"` uses studentized envelope tests to protect for unequal variance, which has performed well in simulations in this context.
`overlay`	logical. For multivariate data, determines whether residuals from different responses are overlaid on a single plot (default), or plotted separately.
`transform`	a character vector pointing to a function that should be applied to both axes of the normal quantile plot. The most common use is to set `transform="pnorm"` for a PP-plot.
`main`	the plot title (if a plot is produced). A vector of three titles, one for each plot. If only one value is given that will be used for all plots.
`xlab`	`x` axis label (if a plot is produced). A vector of three labels, one for each plot. If only one value is given that will be used for all plots.
`ylab`	`y` axis label (if a plot is produced). A vector of three labels, one for each plot. If only one value is given that will be used for all plots.
`col`	color of points
`line.col`	a vector of length three containing the colors of the lines on the three diagnostic plots. Defaults to "slateblue4" for smoothers and to "olivedrab" otherwise. Because it's cool.
`envelope.col`	color of the global envelope around the expected trend. All data points should always stay within this envelope (and will for a proportion `conf.level` of datasets satisfying model assumptions). If a smoother has been used on the residual vs fits or scale-location plot, the envelope is constructed around the smoother, that is, the smoother should always stay within the envelope.

`add.smooth`	logical defaults to `TRUE`, which adds a smoother to residual vs fits and scale-location plots, and computes a global envelope around the *smoother* rather than the data (`add.smooth=FALSE`). No smoother is added to the normal quantile plot.
`plot.it`	logical. Should the result be plotted? If not, a list of analysis outputs is returned, see *Value*.
`resFunction`	the function used to compute residuals for all diagnostic plots. Defaults to `cresiduals` for multivariate linear models, `rstandard` for linear models, or `residuals` in other cases.
`predFunction`	the function used to compute predicted values in residual vs fits or scale-location plots. Defaults to `cpredict` for multivariate linear models and to `cpredict` otherwise.
`fitMin`	the minimum fitted value to use in plots, any fitted value less than this will be truncated to `fitMin`. This is useful for generalised linear models, where small fitted values correspond to predictions that are numerically zero. The default is to set `fitMin` to `-6` for GLMs, otherwise no truncation (`-Inf`).
`...`	further arguments sent through to `plot`.

Details

A challenge when interpreting diagnostic plots is understanding the extent to which deviations from the expected pattern could be due to random noise (sampling variation) rather than actual assumption violations. This function is intended to assess this, by simulating multiple realizations of residuals (and fitted values) in situations where assumptions are satisfied, and plotting a global simulation envelope around these at level `conf.level`.

This function can take any fitted model, and construct any of three diagnostic plots, as determined by `which`:

1. Residual vs fits plot (optionally, with a smoother)
2. Normal quantile plot
3. Scale-Location plot (optionally, with smoother)

and see if the trend is behaving as expected if the model were true. As long as the fitted model responds to `residuals` and `predict` (and when `sim.method="refit"`, `simulate` and `update`) then a simulation envelope will be constructed for each plot.

Simulation envelopes are global, constructed using the `GET-package`, meaning that (for example) a 95% global envelope on a quantile plot should contain *all* residuals for 95% of datasets that satisfy model assumptions. So if *any* data points lie outside the quantile plot's envelope we have evidence that assumptions of the fitted model are not satisfied. The `GET-package` was originally constructed to place envelopes around functions, motivated by the problem of diagnostic testing of spatial processes (Myllymäki et al 2017), but it can equally well be applied here, by treating the set of residuals (ordered according to the x-axis) as point-wise evaluations of a function. For residual vs fits and scale-location plots, if `do.smooth=TRUE`, global envelopes are constructed for the *smoother*, not for the data, hence we are looking to see if the smoother is wholly contained within the envelope. The smoother is constructed using `gam` from the `mgcv` package with maximum likelihood estimation (`method="ML"`).

Note that the global envelopes only tell you if there is evidence of violation of model assumptions – they do not tell you whether the violations are large enough to worry about. For example, in linear models, violations of normality are usually much less important than violations of linearity or equal variance. And in all cases, modest violations tend to only have modest effects on the validity of inferences, so you need to think about how big observed violations are rather than just thinking about whether or not anything is outside its simulation envelope.

The method used to simulate data for the global envelopes is controlled by `sim.method`. The default method (`sim.method="refit"`) uses a *parametric bootstrap* approach: simulate new responses from the fitted model, refit the model and recompute residuals and fitted values. This directly assesses whether trends in observed trends depart from what would be expected if the fitted model were correct, without any further assumptions. For complex models or large datasets this would however be super-slow. A fast, more approximate alternative (`sim.method="norm"`) is to simulate new (multivariate) normal residuals repeatedly and use these to assess whether trends in the observed data depart from what would be expected for independent (multivariate) normal residuals. If residuals are expected to be standard normal, a more refined check is to simulate from the standard normal using (`sim.method="stand.norm"`). This might for example be useful when diagnosing a model fitted using the `mvabund` package (Wang et al. 2012), since this calculates Dunn-Smyth residuals (Dunn & Smyth 1996), which are approximately standard normal when assumptions are satisfied. If `y` is a `glm` with non-Gaussian family then residuals will not be normal and `"refit"` is the only appropriate option, hence other choices will be overridden with a warning reporting that this has been done.

Note that for Multivariate Linear Models (`mlm`), `cresiduals` and `cpredict` are used to construct residuals and fitted values (respectively) from the *full conditional models* (that is, models constructed by regressing each response against all other responses together with predictors). This is done because full conditionals are diagnostic of joint distributions, so *any* violation of multivariate normality is expressed as a violation of linear model assumptions on full conditionals. Results for all responses are overlaid on a single plot, future versions of this function may have an overlay option to separately plot each response.

The simulated data and subsequent analysis are also used to obtain a *P*-value for the test that model assumptions are correct, for each plot. This tests if sample residuals or their smoothers are unusually far from the values expected of them if model assumptions were satisfied. For details see `global_envelope_test`.

Value

Up to three diagnostic plots with simulation envelopes are returned, and additionally a list of three objects used in plotting, for plots 1-3 respectively. Each is a list with five components:

`x` *X*-values used for the envelope. In plots 1 and 3 this is the fitted values, or if `add.smooth=TRUE`, this is 500 equally spaced points covering the range of fitted values. For plot 2, this is sorted normal quantiles corresponding to observed data.

`y` The *Y*-values used for the envelope. In plots 1 and 3 this is the residuals, or with `add.smooth=TRUE`, this is the values of the smoother corresponding to `x`. For plot 2, this is the sorted residuals.

`lo` The lower bound on the global envelope, for each value of `x`.

`hi` The upper bound on the global envelope, for each value of `x`.

`p.value` A *P*-value for the test that observed smoother or data are consistent with what would be expected if the fitted model were correct, computed in `global_envelope_test`.

Author(s)

David Warton <david.warton@unsw.edu.au>

References

Dunn, P. K., & Smyth, G. K. (1996), Randomized quantile residuals. J. Comp. Graphical Stat. 5, 236-244. doi:10.1080/10618600.1996.10474708

Myllymäki, M., Mrkvička, T., Grabarnik, P., Seijo, H. and Hahn, U. (2017), Global envelope tests for spatial processes. J. R. Stat. Soc. B, 79: 381-404. doi:10.1111/rssb.12172

Wang, Y. I., Naumann, U., Wright, S. T., & Warton, D. I. (2012), `mvabund` - an `R` package for model-based analysis of multivariate abundance data. Methods in Ecology and Evolution, 3, 471-474. doi:10.1111/j.2041-210X.2012.00190.x

Warton DI (2022) Eco-Stats - Data Analysis in Ecology, from *t*-tests to multivariate abundances. Springer, ISBN 978-3-030-88442-0

See Also

`cpredict`, `cresiduals`, `qqenvelope`

Examples

```
# fit a Poisson regression to random data:
y = rpois(50,lambda=1)
x = 1:50
rpois_glm = glm(y~x,family=poisson())
plotenvelope(rpois_glm,n.sim=59)

# Fit a multivariate linear model to the iris dataset:
data(iris)
Y = with(iris, cbind(Sepal.Length,Sepal.Width,Petal.Length,Petal.Width))
iris_mlm=lm(Y~Species,data=iris)
# check normality assumption:
plotenvelope(iris_mlm,n.sim=59,which=2)

# A few more plots, with envelopes around data not smoothers:
plotenvelope(iris_mlm, which=1:3, add.smooth=FALSE)
# Note minor violation on the scale/location plot.

# Repeat but with smoothers and with separate plots for each response and
# a multiple testing adjustment to sim envelopes:
plotenvelope(iris_mlm, which=1:3, overlay=FALSE, conf.level=1-0.05/4)
```

qqenvelope — *Normal Quantile-Quantile Plots with Global Simulation Envelopes*

Description

Produces a normal QQ plot of data, or of residuals from a fitted model `y`, with a user-specified line to compare to "theoretical" quantiles, and global envelopes constructed by simulating new residuals. Global envelopes are constructed using the `GET` package for simultaneous control of error rates over the whole curve.

Usage

```
qqenvelope(y, n.sim = 199, conf.level = 0.95, ylab = "Sample Quantiles", ...)
```

Arguments

`y`	can be a set of values for which we wish to check (multivariate) normality, or it can be *any* object that responds to the `residuals`, `simulate` and `update` functions.
`n.sim`	the number of simulated sets of residuals to be generated, to which the observed residuals will be compared. The default is 199 datasets.
`conf.level`	the confidence level to use in constructing the envelope.
`ylab`	`y` axis label (if a plot is produced).
`...`	further arguments sent through to `plotenvelope`

Details

A challenge when interpreting a `qqplot` is understanding the extent to which deviations from expected values could be due to random noise (sampling variation) rather than actual assumption violations. This function is intended to assess this, by simulating multiple realizations of residuals in situations where assumptions are satisfied, and plotting a global (or "simultaneous") simulation envelope around these at level `conf.level`. All data points should lie if assumptions are satisfied, and will do so for a proportion `conf.level` of datasets which satisfy their assumptions.

This function can take data (univariate or multivariate) and check for (multivariate) normality, or it can take a fitted model and use qq plots to interrogate residuals and see if they are behaving as we would expect them to if the model were true.

The envelope is global, constructed using the `GET-package`. So if *any* data points lie outside the envelope we have evidence that assumptions are not satisfied. The `GET-package` was originally constructed to place envelopes around functions, motivated by the problem of diagnostic testing of spatial processes (Myllymäki et al 2017), but it can equally well be applied here, by treating sorted residuals as point-wise evaluations of a function.

For further details refer to `plotenvelope`, which is called to construct the plot.

Value

a qqplot with simulation envelope is returned, and additionally:

`x`	a vector of theoretical quantiles from the standard normal sorted from smallest to largest
`y`	a vector of observed residuals sorted from smallest to largest
`lo`	lower bounds on the global simulation envelope for residuals
`hi`	upper bounds on the global simulation envelope for residuals
`p.value`	A *P*-value for the test that model assumptions are correct, using a 'parametric bootstrap' test, based on how far sample residuals depart from the values expected of them if model assumptions were satisfied.

Author(s)

David Warton <david.warton@unsw.edu.au>

References

Myllymäki, M., Mrkvička, T., Grabarnik, P., Seijo, H. and Hahn, U. (2017), Global envelope tests for spatial processes. J. R. Stat. Soc. B, 79: 381-404. doi:10.1111/rssb.12172 Warton DI (2022) Eco-Stats - Data Analysis in Ecology, from *t*-tests to multivariate abundances. Springer, ISBN 978-3-030-88442-0

See Also

`qqnorm`, `qqline`, `plotenvelope`

Examples

```
# simulate some data and fit a qq plot:
y=rnorm(20)
qqenvelope(y)

# fit a multivariate linear model to the iris dataset:
data(iris)
Y = with(iris, cbind(Sepal.Length,Sepal.Width,Petal.Length,Petal.Width))
iris.mlm=lm(Y~Species,data=iris)
# check normality assumption:
qqenvelope(iris.mlm,n.sim=79)
```

`ravens` *Ravens data*

Description

Data from a study of whether ravens fly towards the sound of gunshots (White 2005). Many thanks to Crow White for providing the raw data. There were twelve locations, at which four treatments were applied (1=gunshot, 2-air horn, 3=whistle, 4=no sound). Ravens within a 100 metre radius of the author were counted ten minutes before and ten minutes after each treatment was applied. Primary interest was in assessing if there was an effect of gunshot on raven numbers. Posthoc analyses suggested that ravens were attracted to gunshots in forested areas only.

Usage

```
data(ravens)
```

Format

A dataframe containing:

Before The number of ravens before the treatment was applied

After The number of ravens after the treatment was applied

delta `After-Before` count

site Location, one of twelve in Jackson Hole, Wyoming's ungulate hunting zone

treatment 1=gunshot, 2=air horn, 3=whistle, 4=no sound

trees 1=forested habitat (>300 trees within 100 metres of observer), 0=open

References

Crow White (2005) Hunters ring dinner bell for ravens: experimental evidence of a unique foraging strategy. Ecology, **86** 1057-60.

Examples

```
data(ravens)
ravens1 = ravens[ravens$treatment==1,]
t.test(ravens1$Before,ravens1$After,paired=TRUE,alternative="less")
boxplot(ravens1$delta,ylab="After-Before counts")
abline(h=0,col="red",lty=3)
```

reveg *Invertebrate abundances in a revegetation study*

Description

Data from a study looking at the effect of revegetation on invertebrate communities (data from Anthony Pik, Macquarie University). Invertebrates were sampled in 4-5 pitfall traps at eight sites that had undergone revegetation, and two sites that hadn't, and it was of interest to see what the effect of revegetation was on the invertebrate community.

Usage

```
data(reveg)
```

Format

A list containing three objects:

abund A data frame with abundances of 24 Orders of invertebrate.

pitfalls A vector specifying the number of pitfall traps that were collected at each site.

treatment Whether the site was a ‘Control‘ or a site that had undergone revegetation (‘Impact‘).

Examples

```
data(reveg)
worms = reveg$abund$Haplotaxida
plot(worms~treatment, data=reveg, horizontal=TRUE,
  las=1, xlab="",ylab="Worm abundance")
```

seaweed *Habitat Configuration data from seaweed experiment*

Description

Data from a study of habitat configuration, specifically, does density of invertebrate epifauna on seaweed vary across sites with different levels of isolation from each other.

Usage

```
data(seaweed)
```

Format

A dataframe containing:

Size A character vector describing size of experimental plots as "SMALL" or "LARGE"

Dist Distance of isolation – 0, 2 or 10 metres from other algal beds

Time Sampling time - either 5 or 10 weeks from the start of the experiment.

Rep The replicate number (1 to 5).

Wmass Wet mass of the algal bed for that plot.

Total Total invertebrate density in the plot, calculated as nuber of individuals divided by `Wmass`.

Other variables in the dataset give invertebrate counts separately for different taxa.

References

Roberts, D. A. & Poore, A. G. (2006). Habitat configuration affects colonisation of epifauna in a marine algal bed. Biological Conservation **127**, 18-26.

Examples

```
data(seaweed)
boxplot(Total~Dist, data=seaweed)
```

`seedsTemp` *Germination rates of* Abutilon angulatum *at different temperatures*

Description

Germination rates of *Abutilon angulatum* from 29 different studies, undertaken at different ambient temperatures. We would like to know how germination rate varies with temperature, in particular, the range of temperatures at which *Abutilon angulatum* will germinate. These data come from a larger study across species and environments to look for a latitudinal signal in tolerance to changing temperature (Sentinella et al 2020).

Usage

```
data(seedsTemp)
```

Format

A dataframe containing:

NumSown The number of seeds sown.

NumGerm The number of seeds that germinated.

Test.Temp The ambient temperature (in degrees Celsius) of the location at which seeds were sown.

References

Sentinella, AT, Warton, DI, Sherwin, WB, Offord, CA, Moles, AT. (2020) Tropical plants do not have narrower temperature tolerances, but are more at risk from warming because they are close to their upper thermal limits. Global Ecol Biogeogr. **29**, 1387-1398.

Examples

```
data(seedsTemp)
seedsTemp$propGerm = seedsTemp$NumGerm / seedsTemp$NumSown
plot(propGerm/(1-propGerm)~Test.Temp,data=seedsTemp,log="y",
 ylab="Germination rate [logit scale]", xlab="Temperature (Celsius)")
```

`simulate.manyglm` *Simulate from manyglm objects*

Description

Simulates new responses for a manyglm object.

Usage

```
## S3 method for class 'manyglm'
simulate(object, nsim = 1, seed = NULL, newdata = object$data, ...)
```

Arguments

`object`	a `manyglm` object from the `mvabund` package.
`nsim`	number of simulated datasets to generate.
`seed`	a seed for random number generation (defaults to NULL)
`newdata`	a new dataset with predictors to simulate new values for. Defaults to data model was fitted to.
`...`	additional optional arguments.

Details

Returns a data frame containing the response and predictors. This function just calls `simulate.cord` from the `ecoCopula` package, on a `cord` object constructed under default settings – that is, it fits a copula latent variable model with two latent variables, then uses this to simulate new data.

Value

Simulates a data frame of new values for responses. If multiple datasets are requested, these are stacked one under the other (see `example(simulate.cord)`.

Author(s)

David Warton <david.warton@unsw.edu.au>

See Also

manyglm, cord, simulate.cord

---

simulate.mlm *Simulate Responses from a Multivariate Linear Model*

Description

Simulate one or more sets of responses from a Multivariate Linear Model (`mlm`) object.

Usage

```
## S3 method for class 'mlm'
simulate(object, nsim = 1, seed = NULL, ...)
```

Arguments

`object`	a `mlm` object, typically the result of calling `lm` where the response is a matrix.
`nsim`	number of replicate datasets to simulate. If `nsim` is greater than 1, the output is arranged in a 3D array.
`seed`	an object specifying if and how the random number generator should be initialized ('seeded'). Either NULL or an integer that will be used in a call to set.seed before simulating the response vectors. If set, the value is saved as the "seed" attribute of the returned value. The default, NULL will not change the random generator state, and return .Random.seed as the "seed" attribute, see 'Value'
`...`	additional optional arguments.

Details

A `simulate` function for `mlm` objects, which simulates one or more sets of responses from a Multivariate Linear Model (`mlm`) object. If multiple sets of responses are requested, they are returned in a 3D array, with simulation number along the third dimension.

The weights argument is currently ignored – a constant variance-covariance matrix assumed for `mlm`.

Value

A matrix of simulated values for the response (or an array, for `nsim` greater than 1)

Author(s)

David Warton <david.warton@unsw.edu.au>

See Also

lm, simulate

Examples

```
# fit a mlm to iris data:
data(iris)
iris.mlm=lm(cbind(Sepal.Length,Sepal.Width,Petal.Length,Petal.Width)~Species,data=iris)
# simulate new responses:
simulate(iris.mlm)
```

`snowmelt` *How flowering time relates to snowmelt date*

Description

Data from a study of how time from snowmelt to flowering relates to snowmelt date and elevation (Wheeler et al 2016) in *Salix herbacea*, based on about 120 plots on each of three mountains in the Swiss Alps.

Usage

```
data(snowmelt)
```

Format

A dataframe containing:

id Plot patch ID

flow Number of days from snowmelt day to flowering

snow Snowmelt day of year

sex Sex of patch, corrected across years, NA - no flowering, 0 - male, 1 - female

elev Elevation (from dGPS)

References

Wheeler, J. A., Cortés, A. J., Sedlacek, J., Karrenberg, S., van Kleunen, M., Wipf, S., Hoch G., Bossdorf, O. & Rixen C. (2016) The snow and the willows: earlier spring snowmelt reduces performance in the low-lying alpine shrub *Salix herbacea*. Journal of Ecology **104**, 1041-50.

Examples

```
data(snowmelt)
plot(flow~snow,data=snowmelt, log="y")
```

`waterQuality` *Water Quality data*

Description

Data from a study of the association between water quality and catchment area. Fish composition were used to construct an index of biotic integrity (IBI) and relate it to catchment are in the Seine river basin in France.

Usage

```
data(waterQuality)
```

Format

A dataframe containing:

catchment The catchment area in square kilometres

quality Index of Biotic Integrity (IBI)

logCatchment log base 10 of catchment area

References

Oberdorff, T. & Hughes, R. M. (1992). Modification of an index of biotic integrity based on fish assemblages to characterize rivers of the Seine Basin, France". Hydrobiologia, **228**, 117-130.

Examples

```
data(waterQuality)
plot(quality~logCatchment, data=waterQuality)
```

`windFarms` *Data from wind farm study*

Description

Data from a study of the effect of offshore wind farms on fish communities (Bergström et al. 2013), with measurements before (2003) and after (2010) wind farm installation at 36 different sites in three zones – two affected by wind farms, and two control zones. Abundances have been recorded for 16 different taxa.

Usage

```
data(windFarms)
```

Format

A list containing three objects:

X A data frame with descriptors of location and time of sampling. These include: ‘Year‘, a factor giving year of sampling, only 2003 and 2010 measurements are available here; ‘Zone‘, a factor giving zone of sampling, `WF` for wind farm, `N` for Northern zone, `S` for Southern zone; ‘Station‘, a factor indicating station ID; ‘Impact‘, a factor indicating whether sampling is ‘Before‘ or ‘After‘ wind farm construction.

abund A data frame containing abundances of 16 different fish taxa.

totalAbund The total abundance of fish at each site.

References

Bergström, L., Sundqvist, F., & Bergström, U. (2013). Effects of an offshore wind farm on temporal and spatial patterns in the demersal fish community. Marine Ecology Progress Series 485, 199-210.

Examples

```
data(windFarms)
eels =windFarms$abund[,14]
plot(eels~interaction(Impact,Zone), data=windFarms$X, horizontal=TRUE,
  las=1, xlab="",ylab="Eel abundance")
```

Index