Package ‘car’

September 18, 2013

Version 2.0-19

Date 2013/09/14

Title Companion to Applied Regression

Depends R (>= 2.14.0), stats, graphics

Imports MASS, nnet

Suggests alr3, boot, leaps, lme4, lmtest, nlme, quantreg, sandwich,mgcv, pbkrtest (>= 0.3-2), rgl (>= 0.93.960), survival, survey

ByteCompile yes

LazyLoad yes

LazyData yes

Description This package accompanies J. Fox and S. Weisberg, An R Companion to Applied Regression,Second Edition, Sage, 2011.

License GPL (>= 2)

URL https://r-forge.r-project.org/projects/car/,http://CRAN.R-project.org/package=car, http://socserv.socsci.mcmaster.ca/jfox/Books/Companion/index.html

Author
John Fox [aut, cre],Sanford Weisberg [aut],Daniel Adler [ctb],Douglas Bates [ctb],Gabriel Baud-Bovy [ctb],Steve Ellison [ctb],David Firth [ctb],Michael Friendly [ctb],Gregor Gorjanc [ctb],Spencer Graves [ctb],Richard Heiberger [ctb],Rafael Laboissiere [ctb],Georges Monette [ctb],Duncan Murdoch [ctb],Henric Nilsson [ctb],Derek Ogle [ctb],Brian Ripley [ctb],William Venables [ctb],Achim Zeileis [ctb],R-Core [ctb]

Maintainer John Fox <jfox@mcmaster.ca>

Repository CRAN

Repository/R-Forge/Project car

Repository/R-Forge/Revision 358

Repository/R-Forge/DateTimeStamp 2013-09-14 18:27:25

Date/Publication 2013-09-18 18:55:33

NeedsCompilation no

R topics documented:

car-package 4
Adler 5
AMSsurvey 5
Angell 6
Anova 7
Anscombe 12
avPlots 13
Baumann 16
bcPower 17
Bfox 18
Blackmoor 19
Boot 20
boxCox 22
boxCoxVariable 24
Boxplot 25
boxTidwell 26
Burt 28
CanPop 28
car-deprecated 29
carWeb 30
ceresPlots 31
Chile 34
Chirot 35
compareCoefs 35
Contrasts 36
Cowles 38
crPlots 39
Davis 41
DavisThin 42
deltaMethod 43
densityPlot 46
Depredations 48
dfbetaPlots 49
Duncan 51
durbinWatsonTest 51
Ellipses 53
Ericksen 56
estimateTransform 57
Florida 59
Freedman 60

Friendly 61
Ginzberg 62
Greene 62
Guyer 63
Hartnagel 64
hccm 65
Highway1 66
hist.boot 67
infIndexPlot 70
influencePlot 71
invResPlot 72
invTranPlot 74
Leinhardt 76
leveneTest 77
leveragePlots 78
linearHypothesis 80
logit 86
Mandel 87
Migration 88
mmps 89
Moore 92
Mroz 93
ncvTest 94
OBrienKaiser 95
Ornstein 96
outlierTest 97
panel.car 99
plot.powerTransform 100
Pottery 101
powerTransform 102
Prestige 104
qqPlot 105
Quartet 108
recode 108
regLine 110
residualPlots 111
Robey 114
Sahlins 115
Salaries 115
scatter3d 116
scatterplot 120
scatterplotMatrix 123
ScatterplotSmoothers 126
showLabels 128
sigmaHat 130
SLID 131
Soils 132
some 133

spreadLevelPlot 134
States 136
subsets 137
symbox 139
testTransform 140
Transact 141
TransformationAxes 142
UN 144
USPop 145
vif 145
Vocab 147
wcrossprod 147
WeightLoss 148
which.names 149
Womenlf 150
Wool 151

Index **152**

`car-package` *Companion to Applied Regression*

Description

This package accompanies Fox, J. and Weisberg, S., *An R Companion to Applied Regression*, Second Edition, Sage, 2011.

Details

Package:	car
Version:	2.0-19
Date:	2013/09/14
Depends:	R (>= 2.14.0), stats, graphics, MASS, nnet
Suggests:	alr3, boot, leaps, lme4, lmtest, nlme, quantreg, sandwich, mgcv, pbkrtest (>= 0.3-2), rgl, survival, survey
License:	GPL (>= 2)
URL:	`http://CRAN.R-project.org/package=car`, `http://socserv.socsci.mcmaster.ca/jfox/Books/Companio`

Author(s)

John Fox <jfox@mcmaster.ca> and Sanford Weisberg. We are grateful to Douglas Bates, Gabriel Baud-Bovy, David Firth, Michael Friendly, Gregor Gorjanc, Spencer Graves, Richard Heiberger, Rafael Laboissiere, Georges Monette, Henric Nilsson, Derek Ogle, Brian Ripley, Achim Zeleis, and R Core for various suggestions and contributions.

Maintainer: John Fox <jfox@mcmaster.ca>

`Adler` *Experimenter Expectations*

Description

The `Adler` data frame has 97 rows and 3 columns.

The “experimenters” were the actual subjects of the study. They collected ratings of the apparent successfulness of people in pictures who were pre-selected for their average appearance. The experimenters were told prior to collecting data that the pictures were either high or low in their appearance of success, and were instructed to get good data, scientific data, or were given no such instruction. Each experimenter collected ratings from 18 randomly assigned respondents; a few subjects were deleted at random to produce an unbalanced design.

Usage

```
Adler
```

Format

This data frame contains the following columns:

instruction a factor with levels: `GOOD`, good data; `NONE`, no stress; `SCIENTIFIC`, scientific data.

expectation a factor with levels: `HIGH`, expect high ratings; `LOW`, expect low ratings.

rating The average rating obtained.

Source

Adler, N. E. (1973) Impact of prior sets given experimenters and subjects on the experimenter expectancy effect. *Sociometry* **36**, 113–126.

References

Erickson, B. H., and Nosanchuk, T. A. (1977) *Understanding Data.* McGraw-Hill Ryerson.

`AMSsurvey` *American Math Society Survey Data*

Description

Counts of new PhDs in the mathematical sciences for 2008-09 and 2011-12 categorized by type of institution, gender, and US citizenship status.

Usage

```
AMSsurvey
```

Format

A data frame with 24 observations on the following 5 variables.

type a factor with levels `I(Pu)` for group I public universities, `I(Pr)` for group I private universities, `II` and `III` for groups II and III, `IV` for statistics and biostatistics programs, and `Va` for applied mathemeatics programs.

sex a factor with levels `Female`, `Male` of the recipient

citizen a factor with levels `Non-US`, `US` giving citizenship status

count The number of individuals of each type in 2008-09

count11 The number of individuals of each type in 2011-12

Details

These data are produced yearly by the American Math Society.

Source

http://www.ams.org/employment/surveyreports.html Supplementary Table 4 in the 2008-09 data.

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Phipps, Polly, Maxwell, James W. and Rose, Colleen (2009), *2009 Annual Survey of the Mathematical Sciences*, 57, 250–259, Supplementary Table 4, http://www.ams/org/employment/2009Survey-First-Report-Supp pdf

---

`Angell` *Moral Integration of American Cities*

Description

The `Angell` data frame has 43 rows and 4 columns. The observations are 43 U. S. cities around 1950.

Usage

```
Angell
```

Format

This data frame contains the following columns:

moral Moral Integration: Composite of crime rate and welfare expenditures.

hetero Ethnic Heterogenity: From percentages of nonwhite and foreign-born white residents.

mobility Geographic Mobility: From percentages of residents moving into and out of the city.

region A factor with levels: `E` Northeast; `MW` Midwest; `S` Southeast; `W` West.

Source

Angell, R. C. (1951) The moral integration of American Cities. *American Journal of Sociology* **57** (part 2), 1–140.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Anova — *Anova Tables for Various Statistical Models*

Description

Calculates type-II or type-III analysis-of-variance tables for model objects produced by `lm`, `glm`, `multinom` (in the **nnet** package), `polr` (in the **MASS** package), `coxph` (in the **survival** package), `lmer` in the **lme4** package, `lme` in the **nlme** package, and for any model with a linear predictor and asymptotically normal coefficients that responds to the `vcov` and `coef` functions. For linear models, F-tests are calculated; for generalized linear models, likelihood-ratio chisquare, Wald chisquare, or F-tests are calculated; for multinomial logit and proportional-odds logit models, likelihood-ratio tests are calculated. Various test statistics are provided for multivariate linear models produced by `lm` or `manova`. Partial-likelihood-ratio tests or Wald tests are provided for Cox models. Wald chi-square tests are provided for fixed effects in linear and generalized linear mixed-effects models. Wald chi-square or F tests are provided in the default case.

Usage

```
Anova(mod, ...)

Manova(mod, ...)

## S3 method for class 'lm'
Anova(mod, error, type=c("II","III", 2, 3),
white.adjust=c(FALSE, TRUE, "hc3", "hc0", "hc1", "hc2", "hc4"),
singular.ok, ...)

## S3 method for class 'aov'
Anova(mod, ...)

## S3 method for class 'glm'
Anova(mod, type=c("II","III", 2, 3),
    test.statistic=c("LR", "Wald", "F"),
    error, error.estimate=c("pearson", "dispersion", "deviance"),
    singular.ok, ...)

## S3 method for class 'multinom'
Anova(mod, type = c("II","III", 2, 3), ...)
```

```
## S3 method for class 'polr'
Anova(mod, type = c("II","III", 2, 3), ...)

## S3 method for class 'mlm'
Anova(mod, type=c("II","III", 2, 3), SSPE, error.df,
    idata, idesign, icontrasts=c("contr.sum", "contr.poly"), imatrix,
    test.statistic=c("Pillai", "Wilks", "Hotelling-Lawley", "Roy"),...)

## S3 method for class 'manova'
Anova(mod, ...)

## S3 method for class 'mlm'
Manova(mod, ...)

## S3 method for class 'Anova.mlm'
print(x, ...)

## S3 method for class 'Anova.mlm'
summary(object, test.statistic, univariate=TRUE,
    multivariate=TRUE, ...)

## S3 method for class 'summary.Anova.mlm'
print(x, digits = getOption("digits"), ... )

## S3 method for class 'coxph'
Anova(mod, type=c("II","III", 2, 3),
test.statistic=c("LR", "Wald"), ...)

## S3 method for class 'lme'
Anova(mod, type=c("II","III", 2, 3),
vcov.=vcov(mod), singular.ok, ...)

## S3 method for class 'mer'
Anova(mod, type=c("II","III", 2, 3),
test.statistic=c("Chisq", "F"), vcov.=vcov(mod), singular.ok, ...)

## S3 method for class 'merMod'
Anova(mod, type=c("II","III", 2, 3),
    test.statistic=c("Chisq", "F"), vcov.=vcov(mod), singular.ok, ...)

## S3 method for class 'svyglm'
Anova(mod, ...)

## Default S3 method:
Anova(mod, type=c("II","III", 2, 3),
test.statistic=c("Chisq", "F"), vcov.=vcov(mod),
singular.ok, ...)
```

Arguments

`mod`	`lm`, `aov`, `glm`, `multinom`, `polr` `mlm`, `coxph`, `lme`, `mer`, `merMod`, `svyglm` or other suitable model object.
`error`	for a linear model, an `lm` model object from which the error sum of squares and degrees of freedom are to be calculated. For F-tests for a generalized linear model, a `glm` object from which the dispersion is to be estimated. If not specified, `mod` is used.
`type`	type of test, `"II"`, `"III"`, `2`, or `3`.
`singular.ok`	defaults to `TRUE` for type-II tests, and `FALSE` for type-III tests (where the tests for models with aliased coefficients will not be straightforwardly interpretable); if `FALSE`, a model with aliased coefficients produces an error.
`test.statistic`	for a generalized linear model, whether to calculate `"LR"` (likelihood-ratio), `"Wald"`, or `"F"` tests; for a Cox model, whether to calculate `"LR"` (partial-likelihood ratio) or `"Wald"` tests; in the default case or for linear mixed models fit by `lmer`, whether to calculate Wald `"Chisq"` or `"F"` tests. For a multivariate linear model, the multivariate test statistic to compute — one of `"Pillai"`, `"Wilks"`, `"Hotelling-Lawley"`, or `"Roy"`, with `"Pillai"` as the default. The `summary` method for `Anova.mlm` objects permits the specification of more than one multivariate test statistic, and the default is to report all four.
`error.estimate`	for F-tests for a generalized linear model, base the dispersion estimate on the Pearson residuals (`"pearson"`, the default); use the dispersion estimate in the model object (`"dispersion"`), which, e.g., is fixed to 1 for binomial and Poisson models; or base the dispersion estimate on the residual deviance (`"deviance"`).
`white.adjust`	if not `FALSE`, the default, tests use a heteroscedasticity-corrected coefficient covariance matrix; the various values of the argument specify different corrections. See the documentation for `hccm` for details. If `white.adjust=TRUE` then the `"hc3"` correction is selected.
`SSPE`	The error sum-of-squares-and-products matrix; if missing, will be computed from the residuals of the model.
`error.df`	The degrees of freedom for error; if missing, will be taken from the model.
`idata`	an optional data frame giving a factor or factors defining the intra-subject model for multivariate repeated-measures data. See *Details* for an explanation of the intra-subject design and for further explanation of the other arguments relating to intra-subject factors.
`idesign`	a one-sided model formula using the "data" in `idata` and specifying the intra-subject design.
`icontrasts`	names of contrast-generating functions to be applied by default to factors and ordered factors, respectively, in the within-subject "data"; the contrasts must produce an intra-subject model matrix in which different terms are orthogonal. The default is `c("contr.sum", "contr.poly")`.
`imatrix`	as an alternative to specifying `idata`, `idesign`, and (optionally) `icontrasts`, the model matrix for the within-subject design can be given directly in the form of list of named elements. Each element gives the columns of the within-subject model matrix for a term to be tested, and must have as many rows as there are

responses; the columns of the within-subject model matrix for different terms must be mutually orthogonal.

`x, object` object of class `"Anova.mlm"` to print or summarize.

`multivariate, univariate`
compute and print multivariate and univariate tests for a repeated-measures ANOVA; the default is `TRUE` for both.

`digits` minimum number of significant digits to print.

`vcov.` an optional coefficient-covariance matrix, computed by default by applying the generic `vcov` function to the model object.

`...` do not use.

Details

The designations "type-II" and "type-III" are borrowed from SAS, but the definitions used here do not correspond precisely to those employed by SAS. Type-II tests are calculated according to the principle of marginality, testing each term after all others, except ignoring the term's higher-order relatives; so-called type-III tests violate marginality, testing each term in the model after all of the others. This definition of Type-II tests corresponds to the tests produced by SAS for analysis-of-variance models, where all of the predictors are factors, but not more generally (i.e., when there are quantitative predictors). Be very careful in formulating the model for type-III tests, or the hypotheses tested will not make sense.

As implemented here, type-II Wald tests are a generalization of the linear hypotheses used to generate these tests in linear models.

For tests for linear models, multivariate linear models, and Wald tests for generalized linear models, Cox models, mixed-effects models, generalized linear models fit to survey data, and in the default case, `Anova` finds the test statistics without refitting the model. The `svyglm` method simply calls the `default` method and therefore can take the same arguments.

The standard R `anova` function calculates sequential ("type-I") tests. These rarely test interesting hypotheses in unbalanced designs.

A MANOVA for a multivariate linear model (i.e., an object of class `"mlm"` or `"manova"`) can optionally include an intra-subject repeated-measures design. If the intra-subject design is absent (the default), the multivariate tests concern all of the response variables. To specify a repeated-measures design, a data frame is provided defining the repeated-measures factor or factors via `idata`, with default contrasts given by the `icontrasts` argument. An intra-subject model-matrix is generated from the formula specified by the `idesign` argument; columns of the model matrix corresponding to different terms in the intra-subject model must be orthogonal (as is insured by the default contrasts). Note that the contrasts given in `icontrasts` can be overridden by assigning specific contrasts to the factors in `idata`. As an alternative, the within-subjects model matrix can be specified directly via the `imatrix` argument. `Manova` is essentially a synonym for `Anova` for multivariate linear models.

Value

An object of class `"anova"`, or `"Anova.mlm"`, which usually is printed. For objects of class `"Anova.mlm"`, there is also a `summary` method, which provides much more detail than the `print` method about the MANOVA, including traditional mixed-model univariate F-tests with Greenhouse-Geisser and Huynh-Feldt corrections.

Warning

Be careful of type-III tests.

Author(s)

John Fox <jfox@mcmaster.ca>; the code for the Mauchly test and Greenhouse-Geisser and Huynh-Feldt corrections for non-spericity in repeated-measures ANOVA are adapted from the functions `stats:::stats:::mauchly.test.SSD` and `stats:::sphericity` by R Core; `summary.Anova.mlm` and `print.summary.Anova.mlm` incorporates code contributed by Gabriel Baud-Bovy.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Hand, D. J., and Taylor, C. C. (1987) *Multivariate Analysis of Variance and Repeated Measures: A Practical Approach for Behavioural Scientists.* Chapman and Hall.

O'Brien, R. G., and Kaiser, M. K. (1985) MANOVA method for analyzing repeated measures designs: An extensive primer. *Psychological Bulletin* **97**, 316–333.

See Also

`linearHypothesis`, `anova` `anova.lm`, `anova.glm`, `anova.mlm`, `anova.coxph`, `link[survey]{svyglm}`.

Examples

```
## Two-Way Anova

mod <- lm(conformity ~ fcategory*partner.status, data=Moore,
  contrasts=list(fcategory=contr.sum, partner.status=contr.sum))
Anova(mod)

## One-Way MANOVA
## See ?Pottery for a description of the data set used in this example.

summary(Anova(lm(cbind(Al, Fe, Mg, Ca, Na) ~ Site, data=Pottery)))

## MANOVA for a randomized block design (example courtesy of Michael Friendly:
##  See ?Soils for description of the data set)

soils.mod <- lm(cbind(pH,N,Dens,P,Ca,Mg,K,Na,Conduc) ~ Block + Contour*Depth,
    data=Soils)
Manova(soils.mod)

## a multivariate linear model for repeated-measures data
## See ?OBrienKaiser for a description of the data set used in this example.

phase <- factor(rep(c("pretest", "posttest", "followup"), c(5, 5, 5)),
    levels=c("pretest", "posttest", "followup"))
hour <- ordered(rep(1:5, 3))
```

```
idata <- data.frame(phase, hour)
idata

mod.ok <- lm(cbind(pre.1, pre.2, pre.3, pre.4, pre.5,
                     post.1, post.2, post.3, post.4, post.5,
                     fup.1, fup.2, fup.3, fup.4, fup.5) ~  treatment*gender,
                data=OBrienKaiser)
(av.ok <- Anova(mod.ok, idata=idata, idesign=~phase*hour))

summary(av.ok, multivariate=FALSE)

## A "doubly multivariate" design with two  distinct repeated-measures variables
## (example courtesy of Michael Friendly)
## See ?WeightLoss for a description of the dataset.

imatrix <- matrix(c(
1,0,-1, 1, 0, 0,
1,0, 0,-2, 0, 0,
1,0, 1, 1, 0, 0,
0,1, 0, 0,-1, 1,
0,1, 0, 0, 0,-2,
0,1, 0, 0, 1, 1), 6, 6, byrow=TRUE)
colnames(imatrix) <- c("WL", "SE", "WL.L", "WL.Q", "SE.L", "SE.Q")
rownames(imatrix) <- colnames(WeightLoss)[-1]
(imatrix <- list(measure=imatrix[,1:2], month=imatrix[,3:6]))
contrasts(WeightLoss$group) <- matrix(c(-2,1,1, 0,-1,1), ncol=2)
(wl.mod<-lm(cbind(wl1, wl2, wl3, se1, se2, se3)~group, data=WeightLoss))
Anova(wl.mod, imatrix=imatrix, test="Roy")

## mixed-effects models examples:

## Not run:
library(nlme)
example(lme)
Anova(fm2)

## End(Not run)

## Not run:
library(lme4)
example(glmer)
Anova(gm1)

## End(Not run)
```

Anscombe *U. S. State Public-School Expenditures*

Description

The Anscombe data frame has 51 rows and 4 columns. The observations are the U. S. states plus Washington, D. C. in 1970.

Usage

```
Anscombe
```

Format

This data frame contains the following columns:

education Per-capita education expenditures, dollars.

income Per-capita income, dollars.

young Proportion under 18, per 1000.

urban Proportion urban, per 1000.

Source

Anscombe, F. J. (1981) *Computing in Statistical Science Through APL*. Springer-Verlag.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

---

avPlots *Added-Variable Plots*

Description

These functions construct added-variable (also called partial-regression) plots for linear and generalized linear models.

Usage

```
avPlots(model, terms=~., intercept=FALSE, layout=NULL, ask, main, ...)

avp(...)

avPlot(model, ...)

## S3 method for class 'lm'
avPlot(model, variable,
id.method = list(abs(residuals(model, type="pearson")), "x"),
labels,
id.n = if(id.method[1]=="identify") Inf else 0,
id.cex=1, id.col=palette()[1],
```

```
col = palette()[1], col.lines = palette()[2],
xlab, ylab, pch = 1, lwd = 2,
main=paste("Added-Variable Plot:", variable),
grid=TRUE,
ellipse=FALSE, ellipse.args=NULL, ...)

## S3 method for class 'glm'
avPlot(model, variable,
id.method = list(abs(residuals(model, type="pearson")), "x"),
labels,
id.n = if(id.method[1]=="identify") Inf else 0,
id.cex=1, id.col=palette()[1],
col = palette()[1], col.lines = palette()[2],
xlab, ylab, pch = 1, lwd = 2,  type=c("Wang", "Weisberg"),
main=paste("Added-Variable Plot:", variable), grid=TRUE,
ellipse=FALSE, ellipse.args=NULL, ...)
```

Arguments

`model`	model object produced by `lm` or `glm`.
`terms`	A one-sided formula that specifies a subset of the predictors. One added-variable plot is drawn for each term. For example, the specification `terms = ~.-X3` would plot against all terms except for `X3`. If this argument is a quoted name of one of the terms, the added-variable plot is drawn for that term only.
`intercept`	Include the intercept in the plots; default is `FALSE`.
`variable`	A quoted string giving the name of a regressor in the model matrix for the horizontal axis
`layout`	If set to a value like `c(1, 1)` or `c(4, 3)`, the layout of the graph will have this many rows and columns. If not set, the program will select an appropriate layout. If the number of graphs exceed nine, you must select the layout yourself, or you will get a maximum of nine per page. If `layout=NA`, the function does not set the layout and the user can use the `par` function to control the layout, for example to have plots from two models in the same graphics window.
`main`	The title of the plot; if missing, one will be supplied.
`ask`	If `TRUE`, ask the user before drawing the next plot; if `FALSE` don't ask.
`...`	`avPlots` passes these arguments to `avPlot`. `avPlot` passes them to `plot`.
`id.method,labels,id.n,id.cex,id.col`	Arguments for the labelling of points. The default is `id.n=0` for labeling no points. See `showLabels` for details of these arguments.
`col`	color for points; the default is the *second* entry in the current color palette (see `palette` and `par`).
`col.lines`	color for the fitted line.
`pch`	plotting character for points; default is `1` (a circle, see `par`).
`lwd`	line width; default is `2` (see `par`).
`xlab`	x-axis label. If omitted a label will be constructed.

ylab	y-axis label. If omitted a label will be constructed.
type	if "Wang" use the method of Wang (1985); if "Weisberg" use the method in the Arc software associated with Cook and Weisberg (1999).
grid	If TRUE, the default, a light-gray background grid is put on the graph.
ellipse	If TRUE, plot a concentration ellipse; default is FALSE.
ellipse.args	Arguments to pass to the link{dataEllipse} function, in the form of a list with named elements; e.g., ellipse.args=list(robust=TRUE)) will cause the ellipse to be plotted using a robust covariance-matrix.

Details

The function intended for direct use is avPlots (for which avp is an abbreviation).

Value

These functions are used for their side effect id producing plots, but also invisibly return the coordinates of the plotted points.

Author(s)

John Fox <jfox@mcmaster.ca>, Sanford Weisberg <sandy@umn.edu>

References

Cook, R. D. and Weisberg, S. (1999) *Applied Regression, Including Computing and Graphics.* Wiley.

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Wang, P C. (1985) Adding a variable in generalized linear models. *Technometrics* **27**, 273–276.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition, Wiley.

See Also

residualPlots, crPlots, ceresPlots, link{dataEllipse}

Examples

```
avPlots(lm(prestige~income+education+type, data=Duncan))

avPlots(glm(partic != "not.work" ~ hincome + children,
  data=Womenlf, family=binomial))
```

Baumann *Methods of Teaching Reading Comprehension*

Description

The `Baumann` data frame has 66 rows and 6 columns. The data are from an experimental study conducted by Baumann and Jones, as reported by Moore and McCabe (1993) Students were randomly assigned to one of three experimental groups.

Usage

```
Baumann
```

Format

This data frame contains the following columns:

group Experimental group; a factor with levels: `Basal`, traditional method of teaching; `DRTA`, an innovative method; `Strat`, another innovative method.

pretest.1 First pretest.

pretest.2 Second pretest.

post.test.1 First post-test.

post.test.2 Second post-test.

post.test.3 Third post-test.

Source

Moore, D. S. and McCabe, G. P. (1993) *Introduction to the Practice of Statistics, Second Edition.* Freeman, p. 794–795.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

bcPower *Box-Cox and Yeo-Johnson Power Transformations*

Description

Transform the elements of a vector using, the Box-Cox, Yeo-Johnson, or simple power transformations.

Usage

```
bcPower(U, lambda, jacobian.adjusted = FALSE)

yjPower(U, lambda, jacobian.adjusted = FALSE)

basicPower(U,lambda)
```

Arguments

U	A vector, matrix or data.frame of values to be transformed
lambda	The one-dimensional transformation parameter, usually in the range from -2 to 2, or if U is a matrix or data frame, a vector of length ncol(U) of transformation parameters
jacobian.adjusted	If TRUE, the transformation is normalized to have Jacobian equal to one. The default is FALSE.

Details

The Box-Cox family of *scaled power transformations* equals $(U^{\lambda} - 1)/\lambda$ for $\lambda \neq 0$, and $\log(U)$ if $\lambda = 0$.

If family="yeo.johnson" then the Yeo-Johnson transformations are used. This is the Box-Cox transformation of $U + 1$ for nonnegative values, and of $|U| + 1$ with parameter $2 - \lambda$ for U negative.

If jacobian.adjusted is TRUE, then the scaled transformations are divided by the Jacobian, which is a function of the geometric mean of U.

The basic power transformation returns U^{λ} if λ is not zero, and $\log(\lambda)$ otherwise.

Missing values are permitted, and return NA where ever Uis equal to NA.

Value

Returns a vector or matrix of transformed values.

Author(s)

Sanford Weisberg, <sandy@umn.edu>

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition. Wiley, Chapter 7.

Yeo, In-Kwon and Johnson, Richard (2000) A new family of power transformations to improve normality or symmetry. *Biometrika*, 87, 954-959.

See Also

powerTransform

Examples

```
U <- c(NA, (-3:3))
## Not run: bcPower(U, 0)  # produces an error as U has negative values
bcPower(U+4,0)
bcPower(U+4, .5, jacobian.adjusted=TRUE)
yjPower(U, 0)
yjPower(U+3, .5, jacobian.adjusted=TRUE)
V <- matrix(1:10, ncol=2)
bcPower(V, c(0,1))
#basicPower(V, c(0,1))
```

Bfox *Canadian Women's Labour-Force Participation*

Description

The Bfox data frame has 30 rows and 7 columns. Time-series data on Canadian women's labor-force participation, 1946–1975.

Usage

```
Bfox
```

Format

This data frame contains the following columns:

partic Percent of adult women in the workforce.

tfr Total fertility rate: expected births to a cohort of 1000 women at current age-specific fertility rates.

menwage Men's average weekly wages, in constant 1935 dollars and adjusted for current tax rates.

womwage Women's average weekly wages.

debt Per-capita consumer debt, in constant dollars.

parttime Percent of the active workforce working 34 hours per week or less.

Warning

The value of `tfr` for 1973 is misrecorded as 2931; it should be 1931.

Source

Fox, B. (1980) *Women's Domestic Labour and their Involvement in Wage Work.* Unpublished doctoral dissertation, p. 449.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

`Blackmoor` *Exercise Histories of Eating-Disordered and Control Subjects*

Description

The `Blackmoor` data frame has 945 rows and 4 columns. Blackmoor and Davis's data on exercise histories of 138 teenaged girls hospitalized for eating disorders and 98 control subjects.

Usage

```
Blackmoor
```

Format

This data frame contains the following columns:

subject a factor with subject id codes.

age age in years.

exercise hours per week of exercise.

group a factor with levels: `control`, Control subjects; `patient`, Eating-disordered patients.

Source

Personal communication from Elizabeth Blackmoor and Caroline Davis, York University.

Boot *Bootstrapping for regression models*

Description

This function provides a simple front-end to the boot function in the package also called boot. Whereas boot is very general and therefore has many arguments, the Boot function has very few arguments, but should meet the needs of many users.

Usage

```
## Default S3 method:
Boot(object, f=coef, labels=names(coef(object)),
                     R=999, method=c("case", "residual"))

## S3 method for class 'lm'
Boot(object, f=coef, labels=names(coef(object)),
                     R=999, method=c("case", "residual"))

## S3 method for class 'glm'
Boot(object, f=coef, labels=names(coef(object)),
                     R=999, method=c("case", "residual"))

## S3 method for class 'nls'
Boot(object, f=coef, labels=names(coef(object)),
                     R=999, method=c("case", "residual"))
```

Arguments

object	A regression object of class lm, glm or nls. The function may work with other regression objects that support the update method and have a subset argument
f	A function whose one argument is the name of a regression object that will be applied to the updated regression object to compute the statistics of interest. The default is coef, to return to regression coefficient estimates. For example, f = function(obj) coef(obj)[1]/coef(obj[2] will bootstrap the ratio of the first and second coefficient estimates.
labels	Provides labels for the statistics computed by f. If this argument is of the wrong length, then generic labels will be generated.
R	Number of bootstrap samples. The number of bootstrap samples actually computed may be smaller than this value if either the fitting method is iterative, or if the rank of a fittle lm or glm model is different in the bootstrap replication than in the original data.
method	The bootstrap method, either "case" for resampling cases or "residuals" for a residual bootstrap. See the details below. The residual bootstrap is available only for lm and nls objects and will return an error for glm objects.

Details

Whereas the boot function is very general, Boot is very specific. It takes the information from a regression object and the choice of method, and creates a function that is passed as the statistic argument to boot. The argument R is also passed to boot. All other arguments to boot are kept at their default values.

The methods available for lm and nls objects are "case" and "residual". The case bootstrap resamples from the joint distribution of the terms in the model and the response. The residual bootstrap fixes the fitted values from the original data, and creates bootstraps by adding a bootstrap sample of the residuals to the fitted values to get a bootstrap response. It is an implementation of Algorithm 6.3, page 271, of Davison and Hinkley (1997). For nls objects ordinary residuals are used in the resampling rather than the standardized residuals used in the lm method. The residual bootstrap for generalized linear models has several competing approaches, but none are without problems. If you want to do a residual bootstrap for a glm, you will need to write your own call to boot.

An attempt to model fit to a bootstrap sample may fail. In a lm or glm fit, the bootstrap sample could have a different rank from the original fit. In an nls fit, convergence may not be obtained for some bootstraps. In either case, NA are returned for the value of the function f. The summary methods handle the NAs appropriately.

Value

See boot for the returned value from this function. The car package includes additional generic functions summary, confint and hist that works with boot objects.

Author(s)

Sanford Weisberg, <sandy@umn.edu>.

References

Davison, A, and Hinkley, D. (1997) *Bootstrap Methods and their Applications*. Oxford: Oxford University Press.

Fox, J. and Weisberg, S. (2011) *Companion to Applied Regression*, Second Edition. Thousand Oaks: Sage.

Fox, J. and Weisberg, S. (2012) *Bootstrapping*, http://socserv.mcmaster.ca/jfox/Books/Companion/appendix/Appendix-Bootstrapping.pdf.

S. Weisberg (2005) *Applied Linear Regression*, Third Edition. Wiley, Chapters 4 and 11.

See Also

Functions that work with Boot objects from the boot package are boot.array, boot.ci, plot.boot and empinf. Additional functions in the car package are summary.boot, confint.boot, and hist.boot.

Examples

```
m1 <- lm(Fertility ~ ., swiss)
betahat.boot <- Boot(m1, R=199) # 199 bootstrap samples--too small to be useful
summary(betahat.boot)  # default summary
```

```
confint(betahat.boot)
hist(betahat.boot)
# Bootstrap for the estimated residual standard deviation:
sigmahat.boot <- Boot(m1, R=199, f=sigmaHat, labels="sigmaHat")
summary(sigmahat.boot)
confint(sigmahat.boot)
```

---

boxCox *Box-Cox Transformations for Linear Models*

Description

Computes and optionally plots profile log-likelihoods for the parameter of the Box-Cox power transformation. This is a slight generalization of the `boxcox` function in the **MASS** package that allows for families of transformations other than the Box-Cox power family.

Usage

```
boxCox(object, ...)

## Default S3 method:
boxCox(object, lambda = seq(-2, 2, 1/10), plotit = TRUE,
       interp = (plotit && (m < 100)), eps = 1/50,
       xlab = expression(lambda),
       ylab = "log-Likelihood", family="bcPower", grid=TRUE, ...)

## S3 method for class 'formula'
boxCox(object, lambda = seq(-2, 2, 1/10), plotit = TRUE,
       interp = (plotit && (m < 100)), eps = 1/50,
       xlab = expression(lambda),
       ylab = "log-Likelihood", family="bcPower", ...)

## S3 method for class 'lm'
boxCox(object, lambda = seq(-2, 2, 1/10), plotit = TRUE,
       interp = (plotit && (m < 100)), eps = 1/50,
       xlab = expression(lambda),
       ylab = "log-Likelihood", family="bcPower", ...)
```

Arguments

`object`	a formula or fitted model object. Currently only `lm` and `aov` objects are handled.
`lambda`	vector of values of lambda, with default (-2, 2) in steps of 0.1, where the profile log-likelihood will be evaluated.
`plotit`	logical which controls whether the result should be plotted; default `TRUE`.
`interp`	logical which controls whether spline interpolation is used. Default to `TRUE` if plotting with lambda of length less than 100.

eps	Tolerance for lambda = 0; defaults to 0.02.
xlab	defaults to "lambda".
ylab	defaults to "log-Likelihood".
family	Defaults to "bcPower" for the Box-Cox power family of transformations. If set to "yjPower" the Yeo-Johnson family, which permits negative responses, is used.
grid	If TRUE, the default, a light-gray background grid is put on the graph.
...	additional parameters to be used in the model fitting.

Details

This routine is an elaboration of the `boxcox` function in the **MASS** package. All arguments except for `family` and `grid` are identical, and if the arguments `family = "bcPower", grid=FALSE` is set it gives an identical graph. If `family = "yjPower"` then the Yeo-Johnson power transformations, which allow nonpositive responses, will be used.

Value

A list of the lambda vector and the computed profile log-likelihood vector, invisibly if the result is plotted. If `plotit=TRUE` plots log-likelihood vs lambda and indicates a 95 lambda. If `interp=TRUE`, spline interpolation is used to give a smoother plot.

Author(s)

Sanford Weisberg, <sandy@umn.edu>

References

Box, G. E. P. and Cox, D. R. (1964) An analysis of transformations. *Journal of the Royal Statisistical Society, Series B*. 26 211-46.

Cook, R. D. and Weisberg, S. (1999) *Applied Regression Including Computing and Graphics*. Wiley.

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition. Wiley.

Yeo, I. and Johnson, R. (2000) A new family of power transformations to improve normality or symmetry. *Biometrika*, 87, 954-959.

See Also

`boxcox`, `yjPower`, `bcPower`, `powerTransform`

Examples

```
boxCox(Volume ~ log(Height) + log(Girth), data = trees,
       lambda = seq(-0.25, 0.25, length = 10))

data("quine", package = "MASS")
boxCox(Days ~ Eth*Sex*Age*Lrn, data = quine,
       lambda = seq(-0.05, 0.45, len = 20), family="yjPower")
```

---

boxCoxVariable *Constructed Variable for Box-Cox Transformation*

Description

Computes a constructed variable for the Box-Cox transformation of the response variable in a linear model.

Usage

```
boxCoxVariable(y)
```

Arguments

y	response variable.

Details

The constructed variable is defined as $y[\log(y/\widetilde{y}) - 1]$, where $\widetilde{y}$ is the geometric mean of `y`.

The constructed variable is meant to be added to the right-hand-side of the linear model. The t-test for the coefficient of the constructed variable is an approximate score test for whether a transformation is required.

If b is the coefficient of the constructed variable, then an estimate of the normalizing power transformation based on the score statistic is $1 - b$. An added-variable plot for the constructed variable shows leverage and influence on the decision to transform `y`.

Value

a numeric vector of the same length as `y`.

Author(s)

John Fox <jfox@mcmaster.ca>

References

Atkinson, A. C. (1985) *Plots, Transformations, and Regression*. Oxford.

Box, G. E. P. and Cox, D. R. (1964) An analysis of transformations. *JRSS B* **26** 211–246.

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

See Also

boxcox, powerTransform, bcPower

Examples

```
mod <- lm(interlocks + 1 ~ assets, data=Ornstein)
mod.aux <- update(mod, . ~ . + boxCoxVariable(interlocks + 1))
summary(mod.aux)
# avPlots(mod.aux, "boxCoxVariable(interlocks + 1)")
```

Boxplot — *Boxplots With Point Identification*

Description

Boxplot is a wrapper for the standard R boxplot function, providing point identification, axis labels, and a formula interface for boxplots without a grouping variable.

Usage

```
Boxplot(y, ...)

## Default S3 method:
Boxplot(y, g, labels, id.method = c("y", "identify", "none"),
    id.n=10, xlab, ylab, ...)

## S3 method for class 'formula'
Boxplot(formula, data = NULL, subset, na.action = NULL, labels.,
    id.method = c("y", "identify", "none"), xlab, ylab, ...)
```

Arguments

y	a numeric variable for which the boxplot is to be constructed.
g	a grouping variable, usually a factor, for constructing parallel boxplots.
labels, labels.	point labels; if not specified, Boxplot will use the row names of the data argument, if one is given, or observation numbers.
id.method	if "y" (the default), all outlying points are labeled; if "identify", points may be labeled interactive; if "none", no point identification is performed.
id.n	up to id.n high outliers and low outliers will be identified in each group, (default, 10).
xlab, ylab	text labels for the horizontal and vertical axes; if missing, Boxplot will use the variable names.
formula	a 'model' formula, of the form ~ y to produce a boxplot for the variable y, or of the form y ~ g, y ~ g1*g2*..., or y ~ g1 + g2 + ... to produce parallel boxplots for y within levels of the grouping variable(s) g, etc., usually factors.

`data, subset, na.action`
: as for statistical modeling functions (see, e.g., `lm`).

`...`
: further arguments, such as `at`, to be passed to `boxplot`.

Author(s)

John Fox <jfox@mcmaster.ca>, with a contribution from Steve Ellison to handle `at` argument (see `boxplot`).

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

See Also

`boxplot`

Examples

```
Boxplot(~income, data=Prestige, id.n=Inf) # identify all outliers
Boxplot(income ~ type, data=Prestige)
Boxplot(income ~ type, data=Prestige, at=c(1, 3, 2))
Boxplot(k5 + k618 ~ lfp*wc, data=Mroz)
with(Prestige, Boxplot(income, labels=rownames(Prestige)))
with(Prestige, Boxplot(income, type, labels=rownames(Prestige)))
```

---

boxTidwell *Box-Tidwell Transformations*

Description

Computes the Box-Tidwell power transformations of the predictors in a linear model.

Usage

```
boxTidwell(y, ...)

## S3 method for class 'formula'
boxTidwell(formula, other.x=NULL, data=NULL, subset,
  na.action=getOption("na.action"), verbose=FALSE, tol=0.001,
  max.iter=25, ...)

## Default S3 method:
boxTidwell(y, x1, x2=NULL, max.iter=25, tol=0.001,
  verbose=FALSE, ...)

## S3 method for class 'boxTidwell'
print(x, digits, ...)
```

Arguments

- `formula` two-sided formula, the right-hand-side of which gives the predictors to be transformed.
- `other.x` one-sided formula giving the predictors that are *not* candidates for transformation, including (e.g.) factors.
- `data` an optional data frame containing the variables in the model. By default the variables are taken from the environment from which `boxTidwell` is called.
- `subset` an optional vector specifying a subset of observations to be used.
- `na.action` a function that indicates what should happen when the data contain `NA`s. The default is set by the `na.action` setting of `options`.
- `verbose` if `TRUE` a record of iterations is printed; default is `FALSE`.
- `tol` if the maximum relative change in coefficients is less than `tol` then convergence is declared.
- `max.iter` maximum number of iterations.
- `y` response variable.
- `x1` matrix of predictors to transform.
- `x2` matrix of predictors that are *not* candidates for transformation.
- `...` not for the user.
- `x` `boxTidwell` object.
- `digits` number of digits for rounding.

Details

The maximum-likelihood estimates of the transformation parameters are computed by Box and Tidwell's (1962) method, which is usually more efficient than using a general nonlinear least-squares routine for this problem. Score tests for the transformations are also reported.

Value

an object of class `boxTidwell`, which is normally just printed.

Author(s)

John Fox `<jfox@mcmaster.ca>`

References

Box, G. E. P. and Tidwell, P. W. (1962) Transformation of the independent variables. *Technometrics* **4**, 531-550.

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Examples

```
boxTidwell(prestige ~ income + education, ~ type + poly(women, 2), data=Prestige)
```

Burt — *Fraudulent Data on IQs of Twins Raised Apart*

Description

The `Burt` data frame has 27 rows and 4 columns. The "data" were simply (and notoriously) manufactured. The same data are in the dataset "twins" in the `alr3` package, but with different labels.

Usage

```
Burt
```

Format

This data frame contains the following columns:

IQbio IQ of twin raised by biological parents

IQfoster IQ of twin raised by foster parents

class A factor with levels (note: out of order): `high`; `low`; `medium`.

Source

Burt, C. (1966) The genetic determination of differences in intelligence: A study of monozygotic twins reared together and apart. *British Journal of Psychology* **57**, 137–153.

CanPop — *Canadian Population Data*

Description

The `CanPop` data frame has 16 rows and 2 columns. Decennial time-series of Canadian population, 1851–2001.

Usage

```
CanPop
```

Format

This data frame contains the following columns:

year census year.

population Population, in millions

Source

Urquhart, M. C. and Buckley, K. A. H. (Eds.) (1965) *Historical Statistics of Canada*. Macmillan, p. 1369.

Canada (1994) *Canada Year Book*. Statistics Canada, Table 3.2.

Statistics Canada: `http://www12.statcan.ca/english/census01/products/standard/popdwell/Table-PR.cfm`.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

`car-deprecated` *Deprecated Functions in car Package*

Description

These functions are provided for compatibility with older versions of the **car** package only, and may be removed eventually. Commands that worked in versions of the **car** package prior to version 2.0-0 will not necessarily work in version 2.0-0 and beyond, or may not work in the same manner.

Usage

```
av.plot(...)
av.plots(...)
box.cox(...)
bc(...)
box.cox.powers(...)
box.cox.var(...)
box.tidwell(...)
cookd(...)
confidence.ellipse(...)
ceres.plot(...)
ceres.plots(...)
cr.plot(...)
cr.plots(...)
data.ellipse(...)
durbin.watson(...)
levene.test(...)
leverage.plot(...)
leverage.plots(...)
linear.hypothesis(...)
ncv.test(...)
outlier.test(...)
qq.plot(...)
scatterplot.matrix(...)
spread.level.plot(...)
```

Arguments

...	pass arguments down.

Details

av.plot and av.plots are now synonyms for the avPlot and avPlots functions.

box.cox and bc are now synonyms for bcPower.

box.cox.powers is now a synonym for powerTransform.

box.cox.var is now a synonym for boxCoxVariable.

box.tidwell is now a synonym for boxTidwell.

cookd is now a synonym for cooks.distance in the **stats** package.

confidence.ellipse is now a synonym for confidenceEllipse.

ceres.plot and ceres.plots are now synonyms for the ceresPlot and ceresPlots functions.

cr.plot and cr.plots are now synonyms for the crPlot and crPlots functions.

data.ellipse is now a synonym for dataEllipse.

durbin.watson is now a synonym for durbinWatsonTest.

levene.test is now a synonym for leveneTest function.

leverage.plot and leverage.plots are now synonyms for the leveragePlot and leveragePlots functions.

linear.hypothesis is now a synonym for the linearHypothesis function.

ncv.test is now a synonym for ncvTest.

outlier.test is now a synonym for outlierTest.

qq.plot is now a synonym for qqPlot.

scatterplot.matrix is now a synonym for scatterplotMatrix.

spread.level.plot is now a synonym for spreadLevelPlot.

---

carWeb — *Access to the R Companion to Applied Regression website*

Description

This function will access the website for *An R Companion to Applied Regression*.

Usage

```
carWeb(page = c("webpage", "errata", "taskviews"), script, data)
```

Arguments

page	A character string indicating what page to open. The default "webpage" will open the main web page, "errata" displays the errata sheet for the book, and "taskviews" fetches and displays a list of available task views from CRAN.
script	The quoted name of a chapter in *An R Companion to Applied Regression*, like "chap-1", "chap-2", up to "chap-8". All the R commands used in that chapter will be displayed in your browser, where you can save them as a text file.
data	The quoted name of a data file in *An R Companion to Applied Regression*, like "Duncan.txt" or "Prestige.txt". The file will be opened in your web browser. You do not need to specify the extension .txt

Value

Either a web page or a PDF document is displayed. Only one of the three arguments page, rfile, or data, should be used.

Author(s)

Sanford Weisberg, based on the function UsingR in the **UsingR** package by John Verzani

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Examples

```
## Not run: carWeb()
```

---

ceresPlots *Ceres Plots*

Description

These functions draw Ceres plots for linear and generalized linear models.

Usage

```
ceresPlots(model, terms = ~., layout = NULL, ask, main,
    ...)

ceresPlot(model, ...)

## S3 method for class 'lm'
ceresPlot(model, variable,
  id.method = list(abs(residuals(model, type="pearson")), "x"),
  labels,
```

```
  id.n = if(id.method[1]=="identify") Inf else 0,
  id.cex=1, id.col=palette()[1],
  line=TRUE, smoother=loessLine, smoother.args=list(),
    smooth, span,
col=palette()[1], col.lines=palette()[-1],
  xlab, ylab, pch=1, lwd=2,
  grid=TRUE, ...)

## S3 method for class 'glm'
ceresPlot(model, ...)
```

Arguments

`model` model object produced by `lm` or `glm`.

`terms` A one-sided formula that specifies a subset of the predictors. One component-plus-residual plot is drawn for each term. The default `~.` is to plot against all numeric predictors. For example, the specification `terms = ~ . - X3` would plot against all predictors except for `X3`. Factors and nonstandard predictors such as B-splines are skipped. If this argument is a quoted name of one of the predictors, the component-plus-residual plot is drawn for that predictor only.

`layout` If set to a value like `c(1, 1)` or `c(4, 3)`, the layout of the graph will have this many rows and columns. If not set, the program will select an appropriate layout. If the number of graphs exceed nine, you must select the layout yourself, or you will get a maximum of nine per page. If `layout=NA`, the function does not set the layout and the user can use the `par` function to control the layout, for example to have plots from two models in the same graphics window.

`ask` If `TRUE`, ask the user before drawing the next plot; if `FALSE`, the default, don't ask. This is relevant only if not all the graphs can be drawn in one window.

`main` Overall title for any array of cerers plots; if missing a default is provided.

`...` `ceresPlots` passes these arguments to `ceresPlot`. `ceresPlot` passes them to `plot`.

`variable` A quoted string giving the name of a variable for the horizontal axis

`id.method,labels,id.n,id.cex,id.col`
Arguments for the labelling of points. The default is `id.n=0` for labeling no points. See `showLabels` for details of these arguments.

`line` `TRUE` to plot least-squares line.

`smoother` Function to add a nonparametric smooth.

`smoother.args` see `ScatterplotSmoothers` for available smooethers and arguments.

`smooth, span` these arguments are included for backwards compatility: if `smooth=TRUE` then `smoother` is set to `loessLine`, and if `span` is specified, it is added to `smoother.args`.

`col` color for points; the default is the first entry in the current color palette (see `palette` and `par`).

`col.lines` a list of at least two colors. The first color is used for the ls line and the second color is used for the fitted lowess line. To use the same color for both, use, for example, `col.lines=c("red", "red")`

`xlab,ylab`	labels for the x and y axes, respectively. If not set appropriate labels are created by the function.
`pch`	plotting character for points; default is `1` (a circle, see `par`).
`lwd`	line width; default is `2` (see `par`).
`grid`	If TRUE, the default, a light-gray background grid is put on the graph

Details

Ceres plots are a generalization of component+residual (partial residual) plots that are less prone to leakage of nonlinearity among the predictors.

The function intended for direct use is `ceresPlots`.

The model cannot contain interactions, but can contain factors. Factors may be present in the model, but Ceres plots cannot be drawn for them.

Value

`NULL`. These functions are used for their side effect: producing plots.

Author(s)

John Fox <jfox@mcmaster.ca>

References

Cook, R. D. and Weisberg, S. (1999) *Applied Regression, Including Computing and Graphics.* Wiley.

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition. Wiley.

See Also

`crPlots`, `avPlots`, `showLabels`

Examples

```
ceresPlots(lm(prestige~income+education+type, data=Prestige), terms= ~ . - type)
```

Chile *Voting Intentions in the 1988 Chilean Plebiscite*

Description

The Chile data frame has 2700 rows and 8 columns. The data are from a national survey conducted in April and May of 1988 by FLACSO/Chile. There are some missing data.

Usage

```
Chile
```

Format

This data frame contains the following columns:

region A factor with levels: C, Central; M, Metropolitan Santiago area; N, North; S, South; SA, city of Santiago.

population Population size of respondent's community.

sex A factor with levels: F, female; M, male.

age in years.

education A factor with levels (note: out of order): P, Primary; PS, Post-secondary; S, Secondary.

income Monthly income, in Pesos.

statusquo Scale of support for the status-quo.

vote a factor with levels: A, will abstain; N, will vote no (against Pinochet); U, undecided; Y, will vote yes (for Pinochet).

Source

Personal communication from FLACSO/Chile.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Chirot — *The 1907 Romanian Peasant Rebellion*

Description

The `Chirot` data frame has 32 rows and 5 columns. The observations are counties in Romania.

Usage

```
Chirot
```

Format

This data frame contains the following columns:

intensity Intensity of the rebellion

commerce Commercialization of agriculture

tradition Traditionalism

midpeasant Strength of middle peasantry

inequality Inequality of land tenure

Source

Chirot, D. and C. Ragin (1975) The market, tradition and peasant rebellion: The case of Romania. *American Sociological Review* **40**, 428–444 [Table 1].

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

compareCoefs — *Print estimated coefficients and their standard errors in a table for several regression models.*

Description

This simple function extracts estimates of regression parameters and their standard errors from one or more models and prints them in a table.

Usage

```
compareCoefs(..., se = TRUE, print=TRUE, digits = 3)
```

Arguments

`...`	One or more regression-model objects. These may be of class `lm`, `glm`, `nlm`, or any other regression method for which the functions `coef` and `vcov` return appropriate values, or if the object inherits from the `mer` class created by the `lme4` package or `lme` in the `nlme` package.
`se`	If `TRUE`, the default, show standard errors as well as estimates, if `FALSE`, show only estimates.
`print`	If `TRUE`, the defualt, the results are printed in a nice format using `printCoefmat`. If `FALSE`, the results are returned as a matrix
`digits`	Passed to the `printCoefmat` function for printing the result.

Value

This function is used for its side-effect of printing the result. It returns a matrix of estimates and standard errors.

Author(s)

John Fox `<jfox@mcmaster.ca>`

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Examples

```
mod1 <- lm(prestige ~ income + education, data=Duncan)
mod2 <- update(mod1, subset=-c(6,16))
mod3 <- update(mod1, . ~ . + type)
compareCoefs(mod1)
compareCoefs(mod1, mod2)
compareCoefs(mod1, mod2, mod3)
compareCoefs(mod1, mod2, se=FALSE)
```

`Contrasts` *Functions to Construct Contrasts*

Description

These are substitutes for similarly named functions in the **stats** package (note the uppercase letter starting the second word in each function name). The only difference is that the contrast functions from the **car** package produce easier-to-read names for the contrasts when they are used in statistical models.

The functions and this documentation are adapted from the **stats** package.

Usage

```
contr.Treatment(n, base = 1, contrasts = TRUE)

contr.Sum(n, contrasts = TRUE)

contr.Helmert(n, contrasts = TRUE)
```

Arguments

`n`	a vector of levels for a factor, or the number of levels.
`base`	an integer specifying which level is considered the baseline level. Ignored if `contrasts` is `FALSE`.
`contrasts`	a logical indicating whether contrasts should be computed.

Details

These functions are used for creating contrast matrices for use in fitting analysis of variance and regression models. The columns of the resulting matrices contain contrasts which can be used for coding a factor with `n` levels. The returned value contains the computed contrasts. If the argument `contrasts` is `FALSE` then a square matrix is returned.

Several aspects of these contrast functions are controlled by options set via the `options` command:

`decorate.contrasts` This option should be set to a 2-element character vector containing the prefix and suffix characters to surround contrast names. If the option is not set, then `c("[", "]")` is used. For example, setting `options(decorate.contrasts=c(".", ""))` produces contrast names that are separated from factor names by a period. Setting `options( decorate.contrasts=c("", ""))` reproduces the behaviour of the R base contrast functions.

`decorate.contr.Treatment` A character string to be appended to contrast names to signify treatment contrasts; if the option is unset, then `"T."` is used.

`decorate.contr.Sum` Similar to the above, with default `"S."`.

`decorate.contr.Helmert` Similar to the above, with default `"H."`.

`contr.Sum.show.levels` Logical value: if `TRUE` (the default if unset), then level names are used for contrasts; if `FALSE`, then numbers are used, as in `contr.sum` in the `base` package.

Note that there is no replacement for `contr.poly` in the `base` package (which produces orthogonal-polynomial contrasts) since this function already constructs easy-to-read contrast names.

Value

A matrix with `n` rows and `k` columns, with `k = n - 1` if `contrasts` is `TRUE` and `k = n` if `contrasts` is `FALSE`.

Author(s)

John Fox <jfox@mcmaster.ca>

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

See Also

`contr.treatment`, `contr.sum`, `contr.helmert`, `contr.poly`

Examples

```
# contr.Treatment vs. contr.treatment in the base package:

lm(prestige ~ (income + education)*type, data=Prestige,
    contrasts=list(type="contr.Treatment"))

##  Call:
##  lm(formula = prestige ~ (income + education) * type, data = Prestige,
##      contrasts = list(type = "contr.Treatment"))
##
##  Coefficients:
##          (Intercept)                  income               education
##             2.275753                0.003522                1.713275
##          type[T.prof]              type[T.wc]      income:type[T.prof]
##            15.351896              -33.536652               -0.002903
##      income:type[T.wc]  education:type[T.prof]    education:type[T.wc]
##            -0.002072                1.387809                4.290875

lm(prestige ~ (income + education)*type, data=Prestige,
    contrasts=list(type="contr.treatment"))

##  Call:
##  lm(formula = prestige ~ (income + education) * type, data = Prestige,
##      contrasts = list(type = "contr.treatment"))
##
##  Coefficients:
##      (Intercept)            income          education
##         2.275753          0.003522           1.713275
##         typeprof            typewc      income:typeprof
##        15.351896        -33.536652          -0.002903
##      income:typewc  education:typeprof    education:typewc
##        -0.002072          1.387809           4.290875
```

`Cowles` *Cowles and Davis's Data on Volunteering*

Description

The `Cowles` data frame has 1421 rows and 4 columns. These data come from a study of the personality determinants of volunteering for psychological research.

Usage

```
Cowles
```

Format

This data frame contains the following columns:

neuroticism scale from Eysenck personality inventory

extraversion scale from Eysenck personality inventory

sex a factor with levels: `female`; `male`

volunteer volunteeing, a factor with levels: `no`; `yes`

Source

Cowles, M. and C. Davis (1987) The subject matter of psychology: Volunteers. *British Journal of Social Psychology* **26**, 97–102.

---

crPlots *Component+Residual (Partial Residual) Plots*

Description

These functions construct component+residual plots (also called partial-residual plots) for linear and generalized linear models.

Usage

```
crPlots(model, terms = ~., layout = NULL, ask, main,
    ...)

crp(...)

crPlot(model, ...)

## S3 method for class 'lm'
crPlot(model, variable,
  id.method = list(abs(residuals(model, type="pearson")), "x"),
  labels,
  id.n = if(id.method[1]=="identify") Inf else 0,
  id.cex=1, id.col=palette()[1],
  order=1, line=TRUE, smoother=loessLine,
smoother.args=list(), smooth, span,
  col=palette()[1], col.lines=palette()[-1],
  xlab, ylab, pch=1, lwd=2, grid=TRUE, ...)
```

Arguments

`model`	model object produced by `lm` or `glm`.
`terms`	A one-sided formula that specifies a subset of the predictors. One component-plus-residual plot is drawn for each term. The default `~.` is to plot against all numeric predictors. For example, the specification `terms = ~ . - X3` would plot against all predictors except for `X3`. If this argument is a quoted name of one of the predictors, the component-plus-residual plot is drawn for that predictor only.
`layout`	If set to a value like `c(1, 1)` or `c(4, 3)`, the layout of the graph will have this many rows and columns. If not set, the program will select an appropriate layout. If the number of graphs exceed nine, you must select the layout yourself, or you will get a maximum of nine per page. If `layout=NA`, the function does not set the layout and the user can use the `par` function to control the layout, for example to have plots from two models in the same graphics window.
`ask`	If `TRUE`, ask the user before drawing the next plot; if `FALSE`, the default, don't ask. This is relevant only if not all the graphs can be drawn in one window.
`main`	The title of the plot; if missing, one will be supplied.
`...`	`crPlots` passes these arguments to `crPlot`. `crPlot` passes them to `plot`.
`variable`	A quoted string giving the name of a variable for the horizontal axis
`id.method,labels,id.n,id.cex,id.col`	Arguments for the labelling of points. The default is `id.n=0` for labeling no points. See `showLabels` for details of these arguments.
`order`	order of polynomial regression performed for predictor to be plotted; default `1`.
`line`	`TRUE` to plot least-squares line.
`smoother`	Function to add a nonparametric smooth.
`smoother.args`	see `ScatterplotSmoothers` for available smooethers and arguments.
`smooth, span`	these arguments are included for backwards compatility: if `smooth=TRUE` then `smoother` is set to `loessLine`, and if `span` is specified, it is added to `smoother.args`.
`col`	color for points; the default is the first entry in the current color palette (see `palette` and `par`).
`col.lines`	a list of at least two colors. The first color is used for the ls line and the second color is used for the fitted lowess line. To use the same color for both, use, for example, `col.lines=c("red", "red")`
`xlab,ylab`	labels for the x and y axes, respectively. If not set appropriate labels are created by the function.
`pch`	plotting character for points; default is `1` (a circle, see `par`).
`lwd`	line width; default is `2` (see `par`).
`grid`	If TRUE, the default, a light-gray background grid is put on the graph

Details

The function intended for direct use is `crPlots`, for which `crp` is an abbreviation.

The model cannot contain interactions, but can contain factors. Parallel boxplots of the partial residuals are drawn for the levels of a factor.

Value

NULL. These functions are used for their side effect of producing plots.

Author(s)

John Fox <jfox@mcmaster.ca>

References

Cook, R. D. and Weisberg, S. (1999) *Applied Regression, Including Computing and Graphics.* Wiley.

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

See Also

ceresPlots, avPlots

Examples

```
crPlots(m<-lm(prestige~income+education, data=Prestige))
# get only one plot
crPlots(m, terms=~ . - education)

crPlots(lm(prestige ~ log2(income) + education + poly(women,2), data=Prestige))

crPlots(glm(partic != "not.work" ~ hincome + children,
  data=Womenlf, family=binomial))
```

Davis *Self-Reports of Height and Weight*

Description

The Davis data frame has 200 rows and 5 columns. The subjects were men and women engaged in regular exercise. There are some missing data.

Usage

```
Davis
```

Format

This data frame contains the following columns:

sex A factor with levels: `F`, female; `M`, male.

weight Measured weight in kg.

height Measured height in cm.

repwt Reported weight in kg.

repht Reported height in cm.

Source

Personal communication from C. Davis, Departments of Physical Education and Psychology, York University.

References

Davis, C. (1990) Body image and weight preoccupation: A comparison between exercising and non-exercising women. *Appetite*, **15**, 13–21.

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

`DavisThin` *Davis's Data on Drive for Thinness*

Description

The `DavisThin` data frame has 191 rows and 7 columns. This is part of a larger dataset for a study of eating disorders. The seven variables in the data frame comprise a "drive for thinness" scale, to be formed by summing the items.

Usage

```
DavisThin
```

Format

This data frame contains the following columns:

DT1 a numeric vector

DT2 a numeric vector

DT3 a numeric vector

DT4 a numeric vector

DT5 a numeric vector

DT6 a numeric vector

DT7 a numeric vector

Source

Davis, C., G. Claridge, and D. Cerullo (1997) Personality factors predisposing to weight preoccupation: A continuum approach to the association between eating disorders and personality disorders. *Journal of Psychiatric Research* **31**, 467–480. [personal communication from the authors.]

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

`deltaMethod` *Estimate and Standard Error of a Nonlinear Function of Estimated Regression Coefficients*

Description

`deltaMethod` is a generic function that uses the delta method to get a first-order approximate standard error for a nonlinear function of a vector of random variables with known or estimated covariance matrix.

Usage

```
deltaMethod(object, ...)

## Default S3 method:
deltaMethod(object, g, vcov., func=g, constants, ...)
## S3 method for class 'lm'
 deltaMethod(object, g, vcov.=vcov,
           parameterNames=names(coef(object)), ...)
## S3 method for class 'nls'
deltaMethod(object, g, vcov.=vcov, ...)
## S3 method for class 'multinom'
 deltaMethod(object, g, vcov. = vcov,
           parameterNames = if (is.matrix(coef(object)))
           colnames(coef(object)) else names(coef(object)), ...)
## S3 method for class 'polr'
 deltaMethod(object, g, vcov.=vcov, ...)
## S3 method for class 'survreg'
 deltaMethod(object, g, vcov. = vcov,
           parameterNames = names(coef(object)), ...)
## S3 method for class 'coxph'
 deltaMethod(object, g, vcov. = vcov,
           parameterNames = names(coef(object)), ...)
## S3 method for class 'mer'
 deltaMethod(object, g, vcov. = vcov,
           parameterNames = names(fixef(object)), ...)
## S3 method for class 'merMod'
 deltaMethod(object, g, vcov. = vcov,
```

```
          parameterNames = names(fixef(object)), ...)
## S3 method for class 'lme'
 deltaMethod(object, g, vcov. = vcov,
          parameterNames = names(fixef(object)), ...)
## S3 method for class 'lmList'
 deltaMethod(object, g,  ...)
```

Arguments

`object`	For the default method, `object` is either (1) a vector of `p` named elements, so `names(object)` returns a list of `p` character strings that are the names of the elements of `object`; or (2) a model object for which there are `coef` and `vcov` methods, and for which the named coefficient vector returned by `coef` is asymptotically normally distributed with asymptotic covariance matrix returned by `vcov`. For the other methods, `object` is a regression object for which `coef(object)` or `fixef(object)` returns a vector of parameter estimates.
`g`	A quoted string that is the function of the parameter estimates to be evaluated; see the details below.
`vcov.`	The (estimated) covariance matrix of the coefficient estimates. For the default method, this argument is required. For all other methods, this argument must either provide the estimated covariance matrix or a function that when applied to `object` returns a covariance matrix. The default is to use the function `vcov`.
`func`	A quoted string used to annotate output. The default of `func = g` is usually appropriate.
`parameterNames`	A character vector of length `p` that gives the names of the parameters in the same order as they appear in the vector of estimates. This argument will be useful if some of the names in the vector of estimates include special characters, like `I(x2^2)`, or `x1:x2` that will confuse the numerical differentiation function. See details below.
`constants`	This argument is a named vector whose elements are constants that are used in the `f` argument. This is needed only when the function is called from within another function to comply to R scoping rules. See example below.
`...`	Used to pass arguments to the generic method.

Details

Suppose x is a random vector of length p that is at least approximately normally distributed with mean β and estimated covariance matrix C. Then any function $g(\beta)$ of β, is estimated by $g(x)$, which is in large samples normally distributed with mean $g(\beta)$ and estimated variance $h'Ch$, where h is the first derivative of $g(\beta)$ with respect to β evaluated at x. This function returns both $g(x)$ and its standard error, the square root of the estimated variance.

The default method requires that you provide x in the argument `object`, C in the argument `vcov.`, and a text expression in argument `g` that when evaluated gives the function g. The call `names(object)` must return the names of the elements of `x` that are used in the expression `g`.

Since the delta method is often applied to functions of regression parameter estimates, the argument object may be the name of a regression object from which the the estimates and their estimated variance matrix can be extracted. In most regression models, estimates are returned by the coef(object) and the variance matrix from vcov(object). You can provide an alternative function for computing the sample variance matrix, for example to use a sandwich estimator.

For mixed models using lme4 or nlme, the coefficient estimates are returned by the fixef function, while for multinom, lmList and nlsList coefficient estimates are returned by coef as a matrix. Methods for these models are provided to get the correct estimates and variance matrix.

The argument g must be a quoted character string that gives the function of interest. For example, if you set m2 <- lm(Y ~ X1 + X2 + X1:X2), then deltaMethod(m2,"X1/X2") applies the delta method to the ratio of the coefficient estimates for X1 and X2. The argument g can consist of constants and names associated with the elements of the vector of coefficient estimates.

In some cases the names may include characters including such as the colon : used in interactions, or mathematical symbols like + or - signs that would confuse the function that computes numerical derivatives, and for this case you can replace the names of the estimates with the parameterNames argument. For example, the ratio of the X2 main effect to the interaction term could be computed using deltaMethod(m2, "b1/b3", parameterNames=c("b0", "b1", "b2", "b3")). The name "(Intercept)" used for the intercept in linear and generalized linear models is an exception, and it will be correctly interpreted by deltaMethod.

For multinom objects, the coef function returns a matrix of coefficients, with each row giving the estimates for comparisons of one category to the baseline. The deltaMethod function applies the delta method to each row of this matrix. Similarly, for lmList and nlsList objects, the delta method is computed for each element of the list of models fit.

For nonlinear regression objects of type nls, the call coef(object) returns the estimated coefficient vectors with names corresponding to parameter names. For example, m2 <- nls(y ~ theta/(1 + gamma * x), start = will have parameters named c("theta", "gamma"). In many other familiar regression methods, such as lm and glm, the names of the coefficient estimates are the corresponding variable names, not parameter names.

For mixed-effects models fit with lmer and nlmer from the lme4 package or lme and nlme from the nlme package, only fixed-effect coefficients are considered.

For regression models for which methods are not provided, you can extract the named vector of coefficient estimates and and estimate of its covariance matrix and then apply the default deltaMethod function.

Earlier versions of deltaMethod included an argument parameterPrefix that implemented the same functionality as the parameterNames argument, but it caused several unintended bugs that were not easily fixed without the change in syntax.

Value

A data.frame with two components named Estimate for the estimate, SE for its standard error. The value of g is given as a row label.

Author(s)

Sanford Weisberg, <sandy@umn.edu>, and John Fox <jfox@mcmaster.ca>

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

S. Weisberg (2005) *Applied Linear Regression*, Third Edition, Wiley, Section 6.1.2.

See Also

First derivatives of g are computed using symbolic differentiation by the function D.

Examples

```
m1 <- lm(time ~ t1 + t2, data = Transact)
deltaMethod(m1, "b1/b2", parameterNames= paste("b", 0:2, sep=""))
deltaMethod(m1, "t1/t2") # use names of preds. rather than coefs.
deltaMethod(m1, "t1/t2", vcov=hccm) # use hccm function to est. vars.
# to get the SE of 1/intercept, rename coefficients
deltaMethod(m1, "1/b0", parameterNames= paste("b", 0:2, sep=""))
# The next example calls the default method by extracting the
# vector of estimates and covariance matrix explicitly
deltaMethod(coef(m1), "t1/t2", vcov.=vcov(m1))
# the following works:
a <- 5
deltaMethod(m1, "(t1 + a)/t2")
# ...but embedded in a function this will fail
f1 <- function(mod, ...) {
 z <- 3
 deltaMethod(m1, "(t1+z)/t2", ...)
 }
## Not run: f1(m1)
# if z is defined globally f1 could even return the wrong answer.
# the following function works
f2 <- function(mod, ...) {
 deltaMethod(m1, "(t1+z)/t2", ...)
 }
f2(m1, constants=c(z=3))
# as does
f3 <- function(mod) {
 a <- 3
 deltaMethod(m1, "(t1+z)/t2", constants=c(z=a))
 }
f3(m1)
```

---

densityPlot *Nonparametric Density Estimates*

Description

densityPlot contructs and graphs nonparametric density estimates, possibly conditioned on a factor. It is a wrapper for the standard R density function.

Usage

```
densityPlot(x, ...)

## Default S3 method:
densityPlot(x, g, bw = "SJ", adjust=1,
    kernel = c("gaussian", "epanechnikov", "rectangular",
               "triangular", "biweight", "cosine", "optcosine"),
    xlab = deparse(substitute(x)), ylab = "Density", col = palette(),
    lty = seq_along(col), lwd = 2, grid=TRUE,
    legend.location = "topright", legend.title = deparse(substitute(g)),
    show.bw = FALSE, rug = TRUE, ...)

## S3 method for class 'formula'
densityPlot(formula, data = NULL, subset, na.action = NULL, xlab, ylab, ...)
```

Arguments

x	a numeric variable, the density of which is estimated.
g	an optional factor to divide the data.
formula	an R model formula, of the form ~ variable to estimate the unconditional density of variable, or variable ~ factor to estimate the density of variable within each level of factor.
data	an optional data frame containing the data.
subset	an optional vector defining a subset of the data.
na.action	a function to handle missing values; defaults to the value of the R na.action option, initially set to na.omit.
bw	the bandwidth for the density estimate(s); either the quoted name of a rule to compute the bandwidth, or a numeric value; the default is "SJ"; if plotting by groups, bw may be a vector of values or rules, one for each group. See density and bw.SJ for details.
adjust	a multiplicative adjustment factor for the bandwidth; the default, 1, indicates no adjustment; if plotting by groups, adjust may be a vector of adjustment factors, one for each group.
kernel	kernel function; the default is "gaussian" (see density).
xlab	label for the horizontal-axis; defaults to the name of the variable x.
ylab	label for the vertical axis; defaults to "Density".
col	vector of colors for the density estimate(s); defaults to the color palette.
lty	vector of line types for the density estimate(s); defaults to the successive integers, starting at 1.
lwd	line width for the density estimate(s); defaults to 2.
grid	if TRUE (the default), grid lines are drawn on the plot.
legend.location	location for the legend when densities are plotted for several groups; defaults to "upperright"; see legend.

`legend.title`	label for the legend, which is drawn if densities are plotted by groups; the default is the name of the factor g.
`show.bw`	if `TRUE`, show the bandwidth(s) in the horizontal-axis label or (for multiple groups) the legend; the default is `FALSE`.
`rug`	if `TRUE` (the default), draw a rug plot (one-dimentional scatterplot) at the bottom of the density estimate.
`...`	arguments to be passed to `plot`.

Value

These functions return `NULL` invisibly and draw graphs.

Author(s)

John Fox `<jfox@mcmaster.ca>`

References

W. N. Venables and B. D. Ripley (2002) *Modern Applied Statistics with S*. New York: Springer.

See Also

`density`, `bw.SJ`, `link{plot.density}`

Examples

```
densityPlot(~ income, show.bw=TRUE, data=Prestige)
densityPlot(income ~ type, data=Prestige)
```

`Depredations` *Minnesota Wolf Depredation Data*

Description

Wolf depredations of livestock on Minnesota farms, 1976-1998.

Usage

```
Depredations
```

Format

A data frame with 434 observations on the following 5 variables.

`longitude` longitude of the farm

`latitude` latitude of the farm

`number` number of depredations 1976-1998

`early` number of depredations 1991 or before

`late` number of depredations 1992 or later

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Harper, Elizabeth K. and Paul, William J. and Mech, L. David and Weisberg, Sanford (2008), Effectiveness of Lethal, Directed Wolf-Depredation Control in Minnesota, *Journal of Wildlife Management*, 72, 3, 778-784. http://pinnacle.allenpress.com/doi/abs/10.2193/2007-273

---

dfbetaPlots *dfbeta and dfbetas Index Plots*

Description

These functions display index plots of dfbeta (effect on coefficients of deleting each observation in turn) and dfbetas (effect on coefficients of deleting each observation in turn, standardized by a deleted estimate of the coefficient standard error). In the plot of dfbeta, horizontal lines are drawn at 0 and +/- one standard error; in the plot of dfbetas, horizontal lines are drawn and 0 and +/- 1.

Usage

```
dfbetaPlots(model, ...)

dfbetasPlots(model, ...)

## S3 method for class 'lm'
dfbetaPlots(model, terms= ~ ., intercept=FALSE, layout=NULL, ask,
    main, xlab, ylab, labels=rownames(dfbeta),
        id.method="y",
        id.n=if(id.method[1]=="identify") Inf else 0, id.cex=1,
        id.col=palette()[1], col=palette()[1], grid=TRUE, ...)

## S3 method for class 'lm'
dfbetasPlots(model, terms=~., intercept=FALSE, layout=NULL, ask,
    main, xlab, ylab,
        labels=rownames(dfbeta), id.method="y",
        id.n=if(id.method[1]=="identify") Inf else 0, id.cex=1,
        id.col=palette()[1], col=palette()[1], grid=TRUE, ...)
```

Arguments

`model`	model object produced by `lm` or `glm`.
`terms`	A one-sided formula that specifies a subset of the terms in the model. One dfbeta or dfbetas plot is drawn for each regressor. The default `~.` is to plot against all terms in the model with the exception of an intercept. For example, the specification `terms = ~.-X3` would plot against all terms except for `X3`. If this argument is a quoted name of one of the terms, the index plot is drawn for that term only.
`intercept`	Include the intercept in the plots; default is `FALSE`.

`layout`	If set to a value like `c(1, 1)` or `c(4, 3)`, the layout of the graph will have this many rows and columns. If not set, the program will select an appropriate layout. If the number of graphs exceed nine, you must select the layout yourself, or you will get a maximum of nine per page. If `layout=NA`, the function does not set the layout and the user can use the `par` function to control the layout, for example to have plots from two models in the same graphics window.
`main`	The title of the graph; if missing, one will be supplied.
`xlab`	Horizontal axis label; defaults to `"Index"`.
`ylab`	Vertical axis label; defaults to coefficient name.
`ask`	If `TRUE`, ask the user before drawing the next plot; if `FALSE`, the default, don't ask.
`...`	optional additional arguments to be passed to `plot`, `points`, and `showLabels`.
`id.method,labels,id.n,id.cex,id.col`	Arguments for the labelling of points. The default is `id.n=0` for labeling no points. See `showLabels` for details of these arguments.
`col`	color for points; defaults to the first entry in the color `palette`.
`grid`	If `TRUE`, the default, a light-gray background grid is put on the graph

Value

`NULL`. These functions are used for their side effect: producing plots.

Author(s)

John Fox <jfox@mcmaster.ca>, Sanford Weisberg <sandy@umn.edu>

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

See Also

`dfbeta` ,`dfbetas`

Examples

```
dfbetaPlots(lm(prestige ~ income + education + type, data=Duncan))

dfbetasPlots(glm(partic != "not.work" ~ hincome + children,
  data=Womenlf, family=binomial))
```

`Duncan` *Duncan's Occupational Prestige Data*

Description

The `Duncan` data frame has 45 rows and 4 columns. Data on the prestige and other characteristics of 45 U. S. occupations in 1950.

Usage

```
Duncan
```

Format

This data frame contains the following columns:

type Type of occupation. A factor with the following levels: `prof`, professional and managerial; `wc`, white-collar; `bc`, blue-collar.

income Percent of males in occupation earning $3500 or more in 1950.

education Percent of males in occupation in 1950 who were high-school graduates.

prestige Percent of raters in NORC study rating occupation as excellent or good in prestige.

Source

Duncan, O. D. (1961) A socioeconomic index for all occupations. In Reiss, A. J., Jr. (Ed.) *Occupations and Social Status.* Free Press [Table VI-1].

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

`durbinWatsonTest` *Durbin-Watson Test for Autocorrelated Errors*

Description

Computes residual autocorrelations and generalized Durbin-Watson statistics and their bootstrapped p-values. `dwt` is an abbreviation for `durbinWatsonTest`.

Usage

```
durbinWatsonTest(model, ...)

dwt(...)

## S3 method for class 'lm'
durbinWatsonTest(model, max.lag=1, simulate=TRUE, reps=1000,
    method=c("resample","normal"),
    alternative=c("two.sided", "positive", "negative"), ...)

## Default S3 method:
durbinWatsonTest(model, max.lag=1, ...)

## S3 method for class 'durbinWatsonTest'
print(x, ...)
```

Arguments

`model`	a linear-model object, or a vector of residuals from a linear model.
`max.lag`	maximum lag to which to compute residual autocorrelations and Durbin-Watson statistics.
`simulate`	if `TRUE` p-values will be estimated by bootstrapping.
`reps`	number of bootstrap replications.
`method`	bootstrap method: `"resample"` to resample from the observed residuals; `"normal"` to sample normally distributed errors with 0 mean and standard deviation equal to the standard error of the regression.
`alternative`	sign of autocorrelation in alternative hypothesis; specify only if `max.lag = 1`; if `max.lag > 1`, then `alternative` is taken to be `"two.sided"`.
`...`	arguments to be passed down.
`x`	`durbinWatsonTest` object.

Value

Returns an object of type `"durbinWatsonTest"`.

Note

p-values are available only from the `lm` method.

Author(s)

John Fox <`jfox@mcmaster.ca`>

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Examples

```
durbinWatsonTest(lm(fconvict ~ tfr + partic + degrees + mconvict, data=Hartnagel))
```

Ellipses *Ellipses, Data Ellipses, and Confidence Ellipses*

Description

These functions draw ellipses, including data ellipses, and confidence ellipses for linear, generalized linear, and possibly other models.

Usage

```
ellipse(center, shape, radius, log="", center.pch=19, center.cex=1.5,
  segments=51, draw=TRUE, add=draw, xlab="", ylab="",
   col=palette()[2], lwd=2, fill=FALSE, fill.alpha=0.3, grid=TRUE, ...)

dataEllipse(x, y, groups, group.labels = group.levels, ellipse.label,
    weights, log = "", levels = c(0.5, 0.95), center.pch = 19,
    center.cex = 1.5, draw = TRUE, plot.points = draw, add = !plot.points,
    segments = 51, robust = FALSE, xlab = deparse(substitute(x)),
    ylab = deparse(substitute(y)),
    col = if (missing(groups)) palette()[1:2] else palette()[1:length(group.levels)],
    pch = if (missing(groups)) 1 else seq(group.levels), lwd = 2,
    fill = FALSE, fill.alpha = 0.3, grid = TRUE, labels, id.method = "mahal",
    id.n = if (id.method[1] == "identify") Inf else 0, id.cex = 1,
    id.col = if (missing(groups)) palette()[1] else palette()(1:length(groups)),
    ...)

confidenceEllipse(model, ...)

## S3 method for class 'lm'
confidenceEllipse(model, which.coef, L, levels=0.95, Scheffe=FALSE, dfn,
  center.pch=19, center.cex=1.5, segments=51, xlab, ylab,
  col=palette()[2], lwd=2, fill=FALSE, fill.alpha=0.3, draw=TRUE, add=!draw, ...)

## S3 method for class 'glm'
confidenceEllipse(model, chisq, ...)

## Default S3 method:
confidenceEllipse(model, which.coef, L, levels=0.95, Scheffe=FALSE,  dfn,
  center.pch=19, center.cex=1.5, segments=51, xlab, ylab,
  col=palette()[2], lwd=2, fill=FALSE, fill.alpha=0.3, draw=TRUE, add=!draw, ...)
```

Arguments

center	2-element vector with coordinates of center of ellipse.
shape	2×2 shape (or covariance) matrix.
radius	radius of circle generating the ellipse.
log	when an ellipse is to be added to an existing plot, indicates whether computations were on logged values and to be plotted on logged axes; "x" if the x-axis is logged, "y" if the y-axis is logged, and "xy" or "yx" if both axes are logged. The default is "", indicating that neither axis is logged.
center.pch	character for plotting ellipse center; if FALSE or NULL the center point isn't plotted.
center.cex	relative size of character for plotting ellipse center.
segments	number of line-segments used to draw ellipse.
draw	if TRUE produce graphical output; if FALSE, only invisibly return coordinates of ellipse(s).
add	if TRUE add ellipse(s) to current plot.
xlab	label for horizontal axis.
ylab	label for vertical axis.
x	a numeric vector, or (if y is missing) a 2-column numeric matrix.
y	a numeric vector, of the same length as x.
groups	optional: a factor to divide the data into groups; a separate ellipse will be plotted for each group (level of the factor).
group.labels	labels to be plotted for the groups; by default, the levels of the groups factor.
ellipse.label	a label for the ellipse(s) or a vector of labels; if several ellipses are drawn and just one label is given, then that label will be repeated. The default is not to label the ellipses.
weights	a numeric vector of weights, of the same length as x and y to be used by cov.wt or cov.trob in computing a weighted covariance matrix; if absent, weights of 1 are used.
plot.points	if FALSE data ellipses are drawn, but points are not plotted.
levels	draw elliptical contours at these (normal) probability or confidence levels.
robust	if TRUE use the cov.trob function in the **MASS** package to calculate the center and covariance matrix for the data ellipse.
model	a model object produced by lm or glm.
which.coef	2-element vector giving indices of coefficients to plot; if missing, the first two coefficients (disregarding the regression constant) will be selected.
L	As an alternative to selecting coefficients to plot, a transformation matrix can be specified to compute two linear combinations of the coefficients; if the L matrix is given, it takes precedence over the which.coef argument. L should have two rows and as many columns as there are coefficients. It can be given directly as a numeric matrix, or specified by a pair of character-valued expressions, in the same manner as for the link{linearHypothesis} function, but with no right-hand side.

`Scheffe`	if `TRUE` scale the ellipse so that its projections onto the axes give Scheffe confidence intervals for the coefficients.
`dfn`	"numerator" degrees of freedom (or just degrees of freedom for a GLM) for drawing the confidence ellipse. Defaults to the number of coefficients in the model (disregarding the constant) if `Scheffe` is `TRUE`, or to `2` otherwise; selecting `dfn = 1` will draw the "confidence-interval generating" ellipse, with projections on the axes corresponding to individual confidence intervals with the stated level of coverage.
`chisq`	if `TRUE`, the confidence ellipse for the coefficients in a generalized linear model are based on the chisquare statistic, if `FALSE` on the F-statistic. This corresponds to using the default and linear-model methods respectively.
`col`	color for lines and ellipse center; the default is the *second* entry in the current color palette (see `palette` and `par`). For `dataEllipse`, two colors can be given, in which case the first is for plotted points and the second for lines and the ellipse center; if ellipses are plotted for `groups`, then this is a vector of colors for the groups.
`pch`	for `dataEllipse` this is the plotting character (default, symbol `1`, a hollow circle) to use for the points; if ellipses are plotted by `groups`, then this a vector of plotting characters, with consecutive symbols starting with `1` as the default.
`lwd`	line width; default is `2` (see `par`).
`fill`	fill the ellipse with translucent color `col` (default, `FALSE`)?
`fill.alpha`	transparency of fill (default = `0.3`).
`...`	other plotting parameters to be passed to `plot` and `line`.
`labels,id.method,id.n,id.cex,id.col`	Arguments for the labelling of points. The default is `id.n=0` for labeling no points. See `showLabels` for details of these arguments.
`grid`	If TRUE, the default, a light-gray background grid is put on the graph

Details

The ellipse is computed by suitably transforming a unit circle.

`dataEllipse` superimposes the normal-probability contours over a scatterplot of the data.

Value

These functions are mainly used for their side effect of producing plots. For greater flexibility (e.g., adding plot annotations), however, `ellipse` returns invisibly the (x, y) coordinates of the calculated ellipse. `dataEllipse` and `confidenceEllipse` return invisibly the coordinates of one or more ellipses, in the latter instance a list named by `levels`.

Author(s)

Georges Monette, John Fox <`jfox@mcmaster.ca`>, and Michael Friendly.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Monette, G. (1990) Geometry of multiple regression and 3D graphics. In Fox, J. and Long, J. S. (Eds.) *Modern Methods of Data Analysis.* Sage.

See Also

cov.trob, cov.wt, linearHypothesis.

Examples

```
dataEllipse(Duncan$income, Duncan$education, levels=0.1*1:9,
    ellipse.label=0.1*1:9, lty=2, fill=TRUE, fill.alpha=0.1)
confidenceEllipse(lm(prestige~income+education, data=Duncan), Scheffe=TRUE)
confidenceEllipse(lm(prestige~income+education, data=Duncan),
L=c("income + education", "income - education"))

wts <- rep(1, nrow(Duncan))
wts[c(6, 16)] <- 0 # delete Minister, Conductor
with(Duncan, {
dataEllipse(income, prestige, levels=0.68)
dataEllipse(income, prestige, levels=0.68, robust=TRUE, plot.points=FALSE, col="green3")
dataEllipse(income, prestige, weights=wts, levels=0.68, plot.points=FALSE, col="brown")
dataEllipse(income, prestige, weights=wts, robust=TRUE, levels=0.68,
plot.points=FALSE, col="blue")
})

with(Prestige, dataEllipse(income, education, type, id.n=2, pch=15:17,
    labels=rownames(Prestige), xlim=c(0, 25000), center.pch="+",
    group.labels=c("Blue Collar", "Professional", "White Collar"),
    ylim=c(5, 20), level=.95, fill=TRUE, fill.alpha=0.1))
```

Ericksen *The 1980 U.S. Census Undercount*

Description

The Ericksen data frame has 66 rows and 9 columns. The observations are 16 large cities, the remaining parts of the states in which these cities are located, and the other U. S. states.

Usage

```
Ericksen
```

Format

This data frame contains the following columns:

minority Percentage black or Hispanic.

crime Rate of serious crimes per 1000 population.

poverty Percentage poor.

language Percentage having difficulty speaking or writing English.

highschool Percentage age 25 or older who had not finished highschool.

housing Percentage of housing in small, multiunit buildings.

city A factor with levels: `city`, major city; `state`, state or state-remainder.

conventional Percentage of households counted by conventional personal enumeration.

undercount Preliminary estimate of percentage undercount.

Source

Ericksen, E. P., Kadane, J. B. and Tukey, J. W. (1989) Adjusting the 1980 Census of Population and Housing. *Journal of the American Statistical Association* **84**, 927–944 [Tables 7 and 8].

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

`estimateTransform` *Finding Univariate or Multivariate Power Transformations*

Description

`estimateTransform` computes members of families of transformations indexed by one parameter, the Box-Cox power family, or the Yeo and Johnson (2000) family, or the basic power family, interpreting zero power as logarithmic. The family can be modified to have Jacobian one, or not, except for the basic power family. Most users will use the function `powerTransform`, which is a front-end for this function.

Usage

```
estimateTransform(X, Y, weights=NULL, family="bcPower", start=NULL,
        method="L-BFGS-B", ...)
```

Arguments

X	A matrix or data.frame giving the “right-side variables”.
Y	A vector or matrix or data.frame giving the “left-side variables.”
weights	Weights as in lm.
family	The transformation family to use. This is the quoted name of a function for computing the transformed values. The default is bcPower for the Box-Cox power family and the most likely alternative is yjPower for the Yeo-Johnson family of transformations.
start	Starting values for the computations. It is usually adequate to leave this at its default value of NULL.
method	The computing alogrithm used by optim for the maximization. The default "L-BFGS-B" appears to work well.
...	Additional arguments that are passed to the optim function that does the maximization. Needed only if there are convergence problems.

Details

See the documentation for the function powerTransform.

Value

An object of class powerTransform with components

value	The value of the loglikelihood at the mle.
counts	See optim.
convergence	See optim.
message	See optim.
hessian	The hessian matrix.
start	Starting values for the computations.
lambda	The ml estimate
roundlam	Convenient rounded values for the estimates. These rounded values will often be the desirable transformations.
family	The transformation family
xqr	QR decomposition of the predictor matrix.
y	The responses to be transformed
x	The predictors
weights	The weights if weighted least squares.

Author(s)

Sanford Weisberg, <sandy@umn.edu>

References

Box, G. E. P. and Cox, D. R. (1964) An analysis of transformations. *Journal of the Royal Statisistical Society, Series B*. 26 211-46.

Cook, R. D. and Weisberg, S. (1999) *Applied Regression Including Computing and Graphics*. Wiley.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Velilla, S. (1993) A note on the multivariate Box-Cox transformation to normality. *Statistics and Probability Letters*, 17, 259-263.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition. Wiley.

Yeo, I. and Johnson, R. (2000) A new family of power transformations to improve normality or symmetry. *Biometrika*, 87, 954-959.

See Also

`powerTransform`, `testTransform`, `optim`.

Examples

```
data(trees,package="MASS")
summary(out1 <- powerTransform(Volume~log(Height)+log(Girth),trees))
# multivariate transformation:
summary(out2 <- powerTransform(cbind(Volume,Height,Girth)~1,trees))
testTransform(out2,c(0,1,0))
# same transformations, but use lm objects
m1 <- lm(Volume~log(Height)+log(Girth),trees)
(out3 <- powerTransform(m1))
# update the lm model with the transformed response
update(m1,basicPower(out3$y,out3$roundlam)~.)
```

`Florida` *Florida County Voting*

Description

The `Florida` data frame has 67 rows and 11 columns. Vote by county in Florida for President in the 2000 election.

Usage

```
Florida
```

Format

This data frame contains the following columns:

GORE Number of votes for Gore

BUSH Number of votes for Bush.

BUCHANAN Number of votes for Buchanan.

NADER Number of votes for Nader.

BROWNE Number of votes for Browne (whoever that is).

HAGELIN Number of votes for Hagelin (whoever that is).

HARRIS Number of votes for Harris (whoever that is).

MCREYNOLDS Number of votes for McReynolds (whoever that is).

MOOREHEAD Number of votes for Moorehead (whoever that is).

PHILLIPS Number of votes for Phillips (whoever that is).

Total Total number of votes.

Source

Adams, G. D. and Fastnow, C. F. (2000) A note on the voting irregularities in Palm Beach, FL. Formerly at `http://madison.hss.cmu.edu/`, but no longer available there.

`Freedman` *Crowding and Crime in U. S. Metropolitan Areas*

Description

The `Freedman` data frame has 110 rows and 4 columns. The observations are U. S. metropolitan areas with 1968 populations of 250,000 or more. There are some missing data.

Usage

```
Freedman
```

Format

This data frame contains the following columns:

population Total 1968 population, 1000s.

nonwhite Percent nonwhite population, 1960.

density Population per square mile, 1968.

crime Crime rate per 100,000, 1969.

Source

United States (1970) *Statistical Abstract of the United States*. Bureau of the Census.

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Freedman, J. (1975) *Crowding and Behavior.* Viking.

`Friendly` *Format Effects on Recall*

Description

The `Friendly` data frame has 30 rows and 2 columns. The data are from an experiment on subjects' ability to remember words based on the presentation format.

Usage

```
Friendly
```

Format

This data frame contains the following columns:

condition A factor with levels: `Before`, Recalled words presented before others; `Meshed`, Recalled words meshed with others; `SFR`, Standard free recall.

correct Number of words correctly recalled, out of 40 on final trial of the experiment.

Source

Friendly, M. and Franklin, P. (1980) Interactive presentation in multitrial free recall. *Memory and Cognition* **8** 265–270 [Personal communication from M. Friendly].

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

`Ginzberg` *Data on Depression*

Description

The `Ginzberg` data frame has 82 rows and 6 columns. The data are for psychiatric patients hospitalized for depression.

Usage

```
Ginzberg
```

Format

This data frame contains the following columns:

simplicity Measures subject's need to see the world in black and white.

fatalism Fatalism scale.

depression Beck self-report depression scale.

adjsimp Adjusted Simplicity: Simplicity adjusted (by regression) for other variables thought to influence depression.

adjfatal Adjusted Fatalism.

adjdep Adjusted Depression.

Source

Personal communication from Georges Monette, Department of Mathematics and Statistics, York University, with the permission of the original investigator.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

`Greene` *Refugee Appeals*

Description

The `Greene` data frame has 384 rows and 7 columns. These are cases filed in 1990, in which refugee claimants rejected by the Canadian Immigration and Refugee Board asked the Federal Court of Appeal for leave to appeal the negative ruling of the Board.

Usage

```
Greene
```

Format

This data frame contains the following columns:

judge Name of judge hearing case. A factor with levels: `Desjardins`, `Heald`, `Hugessen`, `Iacobucci`, `MacGuigan`, `Mahoney`, `Marceau`, `Pratte`, `Stone`, `Urie`.

nation Nation of origin of claimant. A factor with levels: `Argentina`, `Bulgaria`, `China`, `Czechoslovakia`, `El.Salvador`, `Fiji`, `Ghana`, `Guatemala`, `India`, `Iran`, `Lebanon`, `Nicaragua`, `Nigeria`, `Pakistan`, `Poland`, `Somalia`, `Sri.Lanka`.

rater Judgment of independent rater. A factor with levels: `no`, case has no merit; `yes`, case has some merit (leave to appeal should be granted).

decision Judge's decision. A factor with levels: `no`, leave to appeal not granted; `yes`, leave to appeal granted.

language Language of case. A factor with levels: `English`, `French`.

location Location of original refugee claim. A factor with levels: `Montreal`, `other`, `Toronto`.

success Logit of success rate, for all cases from the applicant's nation.

Source

Personal communication from Ian Greene, Department of Political Science, York University.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

`Guyer` *Anonymity and Cooperation*

Description

The `Guyer` data frame has 20 rows and 3 columns. The data are from an experiment in which four-person groups played a prisoner's dilemma game for 30 trails, each person making either a cooperative or competitive choice on each trial. Choices were made either anonymously or in public; groups were composed either of females or of males. The observations are 20 groups.

Usage

```
Guyer
```

Format

This data frame contains the following columns:

cooperation Number of cooperative choices (out of 120 in all).

condition A factor with levels: `A`, Anonymous; `P`, Public-Choice.

sex Sex. A factor with levels: `F`, Female; `M`, Male.

Source

Fox, J. and Guyer, M. (1978) Public choice and cooperation in n-person prisoner's dilemma. *Journal of Conflict Resolution* **22**, 469–481.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

`Hartnagel` *Canadian Crime-Rates Time Series*

Description

The `Hartnagel` data frame has 38 rows and 7 columns. The data are an annual time-series from 1931 to 1968. There are some missing data.

Usage

```
Hartnagel
```

Format

This data frame contains the following columns:

year 1931–1968.

tfr Total fertility rate per 1000 women.

partic Women's labor-force participation rate per 1000.

degrees Women's post-secondary degree rate per 10,000.

fconvict Female indictable-offense conviction rate per 100,000.

ftheft Female theft conviction rate per 100,000.

mconvict Male indictable-offense conviction rate per 100,000.

mtheft Male theft conviction rate per 100,000.

Details

The post-1948 crime rates have been adjusted to account for a difference in method of recording. Some of your results will differ in the last decimal place from those in Table 14.1 of Fox (1997) due to rounding of the data. Missing values for 1950 were interpolated.

Source

Personal communication from T. Hartnagel, Department of Sociology, University of Alberta.

References

Fox, J., and Hartnagel, T. F (1979) Changing social roles and female crime in Canada: A time series analysis. *Canadian Review of Sociology and Anthroplogy*, **16**, 96–104.

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

`hccm` *Heteroscedasticity-Corrected Covariance Matrices*

Description

Calculates heteroscedasticity-corrected covariance matrices linear models fit by least squares or weighted least squares. These are also called “White-corrected” or “White-Huber” covariance matrices.

Usage

```
hccm(model, ...)

## S3 method for class 'lm'
hccm(model, type=c("hc3", "hc0", "hc1", "hc2", "hc4"),
singular.ok=TRUE, ...)

## Default S3 method:
hccm(model, ...)
```

Arguments

`model`	a unweighted or weighted linear model, produced by `lm`.
`type`	one of `"hc0"`, `"hc1"`, `"hc2"`, `"hc3"`, or `"hc4"`; the first of these gives the classic White correction. The `"hc1"`, `"hc2"`, and `"hc3"` corrections are described in Long and Ervin (2000); `"hc4"` is described in Cribari-Neto (2004).
`singular.ok`	if `FALSE` (the default is `TRUE`), a model with aliased coefficients produces an error; otherwise, the aliased coefficients are ignored in the coefficient covariance matrix that's returned.
`...`	arguments to pass to `hccm.lm`.

Details

The classical White-corrected coefficient covariance matrix (`"hc0"`) (for an unweighted model) is

$$V(b) = (X'X)^{-1}X'diag(e_i^2)X(X'X)^{-1}$$

where e_i^2 are the squared residuals, and X is the model matrix. The other methods represent adjustments to this formula. If there are weights, these are incorporated in the corrected covariance amtrix.

The function `hccm.default` simply catches non-`lm` objects.

Value

The heteroscedasticity-corrected covariance matrix for the model.

Author(s)

John Fox <jfox@mcmaster.ca>

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Cribari-Neto, F. (2004) Asymptotic inference under heteroskedasticity of unknown form. *Computational Statistics and Data Analysis* **45**, 215–233.

Long, J. S. and Ervin, L. H. (2000) Using heteroscedasity consistent standard errors in the linear regression model. *The American Statistician* **54**, 217–224. http://www.jstor.org/stable/2685594

White, H. (1980) A heteroskedastic consistent covariance matrix estimator and a direct test of heteroskedasticity. *Econometrica* **48**, 817–838.

Examples

```
options(digits=4)
mod<-lm(interlocks~assets+nation, data=Ornstein)
vcov(mod)
##             (Intercept)     assets  nationOTH   nationUK   nationUS
## (Intercept)   1.079e+00 -1.588e-05 -1.037e+00 -1.057e+00 -1.032e+00
## assets       -1.588e-05  1.642e-09  1.155e-05  1.362e-05  1.109e-05
## nationOTH    -1.037e+00  1.155e-05  7.019e+00  1.021e+00  1.003e+00
## nationUK     -1.057e+00  1.362e-05  1.021e+00  7.405e+00  1.017e+00
## nationUS     -1.032e+00  1.109e-05  1.003e+00  1.017e+00  2.128e+00
hccm(mod)
##             (Intercept)     assets  nationOTH   nationUK   nationUS
## (Intercept)   1.664e+00 -3.957e-05 -1.569e+00 -1.611e+00 -1.572e+00
## assets       -3.957e-05  6.752e-09  2.275e-05  3.051e-05  2.231e-05
## nationOTH    -1.569e+00  2.275e-05  8.209e+00  1.539e+00  1.520e+00
## nationUK     -1.611e+00  3.051e-05  1.539e+00  4.476e+00  1.543e+00
## nationUS     -1.572e+00  2.231e-05  1.520e+00  1.543e+00  1.946e+00
```

Highway1 *Highway Accidents*

Description

The data comes from a unpublished master's paper by Carl Hoffstedt. They relate the automobile accident rate, in accidents per million vehicle miles to several potential terms. The data include 39 sections of large highways in the state of Minnesota in 1973. The goal of this analysis was to understand the impact of design variables, `Acpts`, `Slim`, `Sig`, and `Shld` that are under the control of the highway department, on accidents.

Usage

```
Highway1
```

Format

This data frame contains the following columns:

rate 1973 accident rate per million vehicle miles

len length of the `Highway1` segment in miles

ADT average daily traffic count in thousands

trks truck volume as a percent of the total volume

sigs1 (number of signalized interchanges per mile times len + 1)/len, the number of signals per mile of roadway, adjusted to have no zero values.

slim speed limit in 1973

shld width in feet of outer shoulder on the roadway

lane total number of lanes of traffic

acpt number of access points per mile

itg number of freeway-type interchanges per mile

lwid lane width, in feet

hwy An indicator of the type of roadway or the source of funding for the road, either MC, FAI, PA, or MA

Source

Carl Hoffstedt. This differs from the dataset `highway` in the `alr3` package only by transformation of some of the columns.

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition. Wiley, Section 7.2.

`hist.boot` *Generic functions to provide support for* `boot` *objects*

Description

The `Boot` function in `car` uses the `boot` function from the `boot` package to do a straightforward case or residual bootstrap for a regression object. These are generic functions to summarize the results of the bootstrap.

Usage

```
## S3 method for class 'boot'
hist(x, parm, layout = NULL, ask, main = "", freq = FALSE,
    estPoint = TRUE, point.col = "black", point.lty = 2, point.lwd = 2,
    estDensity = !freq, den.col = "blue", den.lty = 1, den.lwd = 2,
    estNormal = !freq, nor.col = "red", nor.lty = 2, nor.lwd = 2,
    ci = c("bca", "none", "percentile"), level = 0.95, legend = c("top",
        "none", "separate"), box = TRUE, ...)

## S3 method for class 'boot'
summary(object, parm, high.moments = FALSE, extremes = FALSE, ...)

## S3 method for class 'boot'
confint(object, parm, level = 0.95, type = c("bca", "norm",
    "basic", "perc", "all"), ...)
```

Arguments

`x, object` An object created by a call to `boot` of class `"boot"`.

`parm` A vector of numbers or coefficient names giving the coefficients for which a histogram or confidence interval is desired. If numbers are used, 1 corresponds to the intercept, if any. The default is all coefficients.

`layout` If set to a value like `c(1, 1)` or `c(4, 3)`, the layout of the graph will have this many rows and columns. If not set, the program will select an appropriate layout. If the number of graphs exceed nine, you must select the layout yourself, or you will get a maximum of nine per page. If `layout=NA`, the function does not set the layout and the user can use the `par` function to control the layout, for example to have plots from two models in the same graphics window.

`ask` If `TRUE`, ask the user before drawing the next plot; if `FALSE`, don't ask.

`main` Main title for the graphs. The default is `main=""` for no title.

`freq` The usual default for `hist` is `freq=TRUE` to give a frequency histogram. The default here is `freq=FALSE` to give a density histogram. A density estimate and/or a fitted normal density can be added to the graph if `freq=FALSE` but not if `freq=TRUE`.

`estPoint, point.col, point.lty, point.lwd`
If `estPoint=TRUE`, the default, a vertical line is drawn on the histgram at the value of the point estimate computed from the complete data. The remaining three optional arguments set the color, line type and line width of the line that is drawn.

`estDensity, den.col, den.lty, den.lwd`
If `estDensity=TRUE` and`freq=FALSE`, the default, a kernel density estimate is drawn on the plot with a call to the `density` function with no additional arguments. The remaining three optional arguments set the color, line type and line width of the lines that are drawn.

`estNormal, nor.col, nor.lty, nor.lwd`
: If `estNormal=TRUE` and`freq=FALSE`, the default, a normal density with mean and sd computed from the data is drawn on the plot. The remaining three optional arguments set the color, line type and line width of the lines that are drawn.

`ci`
: A confidence interval based on the bootstrap will be added to the histogram using the BCa method if `ci="bca"` or using the percentile method if `ci="percentile"`. No interval is drawn if `ci="none"`. The default is `"bca"`. The interval is indicated by a thick horizontal line at `y=0`. For some bootstraps the BCa method is unavailable and another method should be used.

`legend`
: A legend can be added to the (array of) histograms. The value “"top"” puts at the top-left of the plots. The value “"separate"” puts the legend in its own graph following all the histograms. The value “"none"” suppresses the legend.

`box`
: Add a box around each histogram.

`...`
: Additional arguments passed to `hist`; for other methods this is included for compatibility with the generic method. For example, the argument `border=par()$bg` in `hist` will draw the histogram transparently, leaving only the density estimates.

`high.moments`
: Should the skewness and kurtosis be included in the summary? Default is FALSE.

`extremes`
: Should the minimum, maximum and range be included in the summary? Default is FALSE.

`level`
: Confidence level, a number between 0 and 1. In `confint`, `level` can be a vector; for example `level=c(.68, .90, .95)` will return the estimated quantiles at `c(.025, .05, .16, .84, .95, .975)`.

`type`
: Selects the confidence interval type. The types implemented are the `"percentile"` method, which uses the function `quantile` to return the appropriate quantiles for the confidence limit specified, the default `bca` which uses the bias-corrected and accelerated method presented by Efron and Tibshirani (1993, Chapter 14). For the other types, see the documentation for `boot`.

Value

`hist` is used for the side-effect of drawing an array of historgams of each column of the first argument. `summary` returns a matrix of summary statistics for each of the columns in the bootstrap object. The `confint` method returns confidence intervals. Print method

Author(s)

Sanford Weisberg, <sandy@umn.edu>

References

Efron, B. and Tibsharini, R. (1993) *An Introduction to the Bootstrap*. New York: Chapman and Hall.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2012) *Bootstrapping*, http://socserv.mcmaster.ca/jfox/Books/Companion/appendix/Appendix-Bootstrapping.pdf.

Weisberg, S. (2013) *Applied Linear Regression*, Fourth Edition, Wiley

See Also

See Also `Boot`, `hist`, `density`

Examples

```
m1 <- lm(Fertility ~ ., swiss)
betahat.boot <- Boot(m1, R=99) # 99 bootstrap samples--too small to be useful
summary(betahat.boot)  # default summary
confint(betahat.boot)
hist(betahat.boot)
```

`infIndexPlot` *Influence Index Plot*

Description

Provides index plots of Cook's distances, leverages, Studentized residuals, and outlier significance levels for a regression object.

Usage

```
infIndexPlot(model, ...)

influenceIndexPlot(model, ...)

## S3 method for class 'lm'
infIndexPlot(model,
     vars=c("Cook", "Studentized", "Bonf", "hat"),
     main="Diagnostic Plots",
     labels, id.method = "y",
     id.n = if(id.method[1]=="identify") Inf else 0,
     id.cex=1, id.col=palette()[1], grid=TRUE, ...)
```

Arguments

`model` A regression object of class `lm` or `glm`.

`vars` All the quantities listed in this argument are plotted. Use `"Cook"` for Cook's distances, `"Studentized"` for Studentized residuals, `"Bonf"` for Bonferroni p-values for an outlier test, and and `"hat"` for hat-values (or leverages). Capitalization is optional. All may be abbreviated by the first one or more letters.

`main` main title for graph

`id.method,labels,id.n,id.cex,id.col`
Arguments for the labelling of points. The default is `id.n=0` for labeling no points. See `showLabels` for details of these arguments.

grid	If TRUE, the default, a light-gray background grid is put on the graph
...	Arguments passed to plot

Value

Used for its side effect of producing a graph. Produces four index plots of Cook's distance, Studentized Residuals, the corresponding Bonferroni p-values for outlier tests, and leverages.

Author(s)

Sanford Weisberg, <sandy@umn.edu>

References

Cook, R. D. and Weisberg, S. (1999) *Applied Regression, Including Computing and Graphics.* Wiley.

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition. Wiley.

See Also

cooks.distance, rstudent, outlierTest, hatvalues

Examples

```
m1 <- lm(prestige ~ income + education + type, Duncan)
influenceIndexPlot(m1)
```

---

influencePlot *Regression Influence Plot*

Description

This function creates a "bubble" plot of Studentized residuals by hat values, with the areas of the circles representing the observations proportional to Cook's distances. Vertical reference lines are drawn at twice and three times the average hat value, horizontal reference lines at -2, 0, and 2 on the Studentized-residual scale.

Usage

```
influencePlot(model, ...)

## S3 method for class 'lm'
influencePlot(model, scale=10,
xlab="Hat-Values", ylab="Studentized Residuals",
    labels, id.method = "noteworthy",
    id.n = if(id.method[1]=="identify") Inf else 0,
    id.cex=1, id.col=palette()[1], ...)
```

Arguments

- `model` a linear or generalized-linear model.
- `scale` a factor to adjust the size of the circles.
- `xlab, ylab` axis labels.
- `labels, id.method, id.n, id.cex, id.col`
  settings for labelling points; see `link{showLabels}` for details. To omit point labelling, set `id.n=0`, the default. The default `id.method="noteworthy"` is used only in this function and indicates setting labels for points with large Studentized residuals, hat-values or Cook's distances. Set `id.method="identify"` for interactive point identification.
- `...` arguments to pass to the `plot` and `points` functions.

Value

If points are identified, returns a data frame with the hat values, Studentized residuals and Cook's distance of the identified points. If no points are identified, nothing is returned. This function is primarily used for its side-effect of drawing a plot.

Author(s)

John Fox <jfox@mcmaster.ca>, minor changes by S. Weisberg <sandy@umn.edu>

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

See Also

`cooks.distance`, `rstudent`, `hatvalues`, `showLabels`

Examples

```
influencePlot(lm(prestige ~ income + education, data=Duncan))
```

---

invResPlot *Inverse Response Plots to Transform the Response*

Description

For a `lm` model, draws an inverse.response plot with the response Y on the vertical axis and the fitted values $\hat{Y}$ on the horizontal axis. Uses `nls` to estimate λ in the function $\hat{Y} = b_0 + b_1 Y^{\lambda}$. Adds the fitted curve to the plot. `invResPlot` is an alias for `inverseResponsePlot`.

Usage

```
inverseResponsePlot(model, lambda=c(-1,0,1), robust=FALSE, xlab=NULL, ...)

## S3 method for class 'lm'
inverseResponsePlot(model, lambda=c(-1,0,1), robust=FALSE,
   xlab=NULL, labels=names(residuals(model)), ...)

invResPlot(model, ...)
```

Arguments

`model`	A `lm` regression object
`lambda`	A vector of values for lambda. A plot will be produced with curves corresponding to these lambdas and to the least squares estimate of lambda
`xlab`	The horizontal axis label. If `NULL`, it is constructed by the function.
`labels`	Case labels if labeling is turned on; see `invTranPlot` and `showLabels` for arguments.
`robust`	If TRUE, then estimation uses Huber M-estimates with the median absolute deviation to estimate scale and k= 1.345. The default is FALSE.
`...`	Other arguments passed to `invTranPlot` and then to `plot`.

Value

As a side effect, a plot is produced with the response on the horizontal axis and fitted values on the vertical axis. Several lines are added to be plot as the ols estimates of the regression of $\hat{Y}$ on Y^{λ}, interpreting $\lambda = 0$ to be natural logarithms.

Numeric output is a list with elements

`lambda`	Estimate of transformation parameter for the response
`RSS`	The residual sum of squares at the minimum if robust=FALSE. If robust = TRUE, the value of Huber objective function is returned.

Author(s)

Sanford Weisberg, `sandy@umn.edu`

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Pendergast, L, and Sheather, S. (in press). On sensitivity of response plot estimation of a robust estimation approach. *Scandinavian Journal of Statistics*.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition, Wiley, Chapter 7.

See Also

`invTranPlot`, `powerTransform`, `showLabels`

Examples

```
m2 <- lm(rate ~ log(len) + log(ADT) + slim + shld + log(sigs1), Highway1)
invResPlot(m2)
```

---

invTranPlot *Choose a Predictor Transformation Visually or Numerically*

Description

invTranPlot draws a two-dimensional scatterplot of Y versus X, along with the OLS fit from the regression of Y on $(X^\lambda - 1)/\lambda$. invTranEstimate finds the nonlinear least squares estimate of λ and its standard error.

Usage

```
invTranPlot(x, ...)

## S3 method for class 'formula'
invTranPlot(x, data, subset, na.action, ...)

## Default S3 method:
invTranPlot(x, y, lambda=c(-1, 0, 1), robust=FALSE,
        lty.lines=rep(c("solid", "dashed", "dotdash", "longdash", "twodash"),
        length=1 + length(lambda)), lwd.lines=2,
        col=palette()[1], col.lines=palette(),
        xlab=deparse(substitute(x)), ylab=deparse(substitute(y)),
        family="bcPower", optimal=TRUE, key="auto",
        id.method = "x",
        labels,
        id.n = if(id.method[1]=="identify") Inf else 0,
        id.cex=1, id.col=palette()[1], grid=TRUE, ...)

invTranEstimate(x, y, family="bcPower", confidence=0.95, robust=FALSE)
```

Arguments

x	The predictor variable, or a formula with a single response and a single predictor
y	The response variable
data	An optional data frame to get the data for the formula
subset	Optional, as in lm, select a subset of the cases
na.action	Optional, as in lm, the action for missing data
lambda	The powers used in the plot. The optimal power than minimizes the residual sum of squares is always added unless optimal is FALSE.

`robust`	If TRUE, then the estimated transformation is computed using Huber M-estimation with the MAD used to estimate scale and k=1.345. The default is FALSE.
`family`	The transformation family to use, `"bcPower"`, `"yjPower"`, or a user-defined family.
`confidence`	returns a profile likelihood confidence interval for the optimal transformation with this confidence level. If `FALSE`, or if `robust=TRUE`, no interval is returned.
`optimal`	Include the optimal value of lambda?
`lty.lines`	line types corresponding to the powers
`lwd.lines`	the width of the plotted lines, defaults to 2 times the standard
`col`	color(s) of the points in the plot. If you wish to distinguish points according to the levels of a factor, we recommend using symbols, specified with the `pch` argument, rather than colors.
`col.lines`	color of the fitted lines corresponding to the powers. The default is to use the colors returned by `palette`
`key`	The default is `"auto"`, in which case a legend is added to the plot, either above the top marign or in the bottom right or top right corner. Set to NULL to suppress the legend.
`xlab`	Label for the horizontal axis.
`ylab`	Label for the vertical axis.
`id.method,labels,id.n,id.cex,id.col`	Arguments for the labelling of points. The default is `id.n=0` for labeling no points. See `showLabels` for details of these arguments.
`...`	Additional arguments passed to the plot method, such as `pch`.
`grid`	If TRUE, the default, a light-gray background grid is put on the graph

Value

`invTranPlot` plots a graph and returns a data frame with λ in the first column, and the residual sum of squares from the regression for that λ in the second column.

`invTranEstimate` returns a list with elements `lambda` for the estimate, `se` for its standard error, and `RSS`, the minimum value of the residual sum of squares.

Author(s)

Sanford Weisberg, `<sandy@umn.edu>`

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Pendergast, L, and Sheather, S. (in press). On sensitivity of response plot estimation of a robust estimation approach. *Scandinavian Journal of Statistics*.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition. Wiley.

See Also

inverseResponsePlot,optimize

Examples

```
with(UN, invTranPlot(gdp, infant.mortality))
with(UN, invTranEstimate(gdp, infant.mortality))
```

Leinhardt *Data on Infant-Mortality*

Description

The Leinhardt data frame has 105 rows and 4 columns. The observations are nations of the world around 1970.

Usage

```
Leinhardt
```

Format

This data frame contains the following columns:

income Per-capita income in U. S. dollars.

infant Infant-mortality rate per 1000 live births.

region A factor with levels: Africa; Americas; Asia, Asia and Oceania; Europe.

oil Oil-exporting country. A factor with levels: no, yes.

Details

The infant-mortality rate for Jamaica is misprinted in Leinhardt and Wasserman; the correct value is given here. Some of the values given in Leinhardt and Wasserman do not appear in the original New York Times table and are of dubious validity.

Source

Leinhardt, S. and Wasserman, S. S. (1979) Exploratory data analysis: An introduction to selected methods. In Schuessler, K. (Ed.) *Sociological Methodology 1979* Jossey-Bass.

The New York Times, 28 September 1975, p. E-3, Table 3.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

leveneTest *Levene's Test*

Description

Computes Levene's test for homogeneity of variance across groups.

Usage

```
leveneTest(y, ...)
## S3 method for class 'formula'
leveneTest(y, data, ...)
## S3 method for class 'lm'
leveneTest(y, ...)
## Default S3 method:
leveneTest(y, group, center=median, ...)
```

Arguments

`y`	response variable for the default method, or a `lm` or `formula` object. If `y` is a linear-model object or a formula, the variables on the right-hand-side of the model must all be factors and must be completely crossed.
`group`	factor defining groups.
`center`	The name of a function to compute the center of each group; `mean` gives the original Levene's test; the default, `median`, provides a more robust test.
`data`	a data frame for evaluating the `formula`.
`...`	arguments to be passed down, e.g., `data` for the `formula` and `lm` methods; can also be used to pass arguments to the function given by `center` (e.g., `center=mean` and `trim=0.1` specify the 10% trimmed mean).

Value

returns an object meant to be printed showing the results of the test.

Note

adapted from a response posted by Brian Ripley to the r-help email list.

Author(s)

John Fox <`jfox@mcmaster.ca`>; original generic version contributed by Derek Ogle

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Examples

```
with(Moore, leveneTest(conformity, fcategory))
with(Moore, leveneTest(conformity, interaction(fcategory, partner.status)))
leveneTest(conformity ~ fcategory*partner.status, data=Moore)
leveneTest(lm(conformity ~ fcategory*partner.status, data=Moore))
leveneTest(conformity ~ fcategory*partner.status, data=Moore, center=mean)
leveneTest(conformity ~ fcategory*partner.status, data=Moore, center=mean, trim=0.1)
```

`leveragePlots` *Regression Leverage Plots*

Description

These functions display a generalization, due to Sall (1990) and Cook and Weisberg (1991), of added-variable plots to multiple-df terms in a linear model. When a term has just 1 df, the leverage plot is a rescaled version of the usual added-variable (partial-regression) plot.

Usage

```
leveragePlots(model, terms = ~., layout = NULL, ask,
    main, ...)

leveragePlot(model, ...)

## S3 method for class 'lm'
leveragePlot(model, term.name,
    id.method = list(abs(residuals(model, type="pearson")), "x"),
    labels,
    id.n = if(id.method[1]=="identify") Inf else 0,
    id.cex=1, id.col=palette()[1],
    col=palette()[1], col.lines=palette()[2], lwd=2,
    xlab, ylab, main="Leverage Plot", grid=TRUE, ...)

## S3 method for class 'glm'
leveragePlot(model, ...)
```

Arguments

`model` model object produced by `lm`

`terms` A one-sided formula that specifies a subset of the predictors. One added-variable plot is drawn for each term. The default `~.` is to plot against all numeric predictors. For example, the specification `terms = ~ . - X3` would plot against all predictors except for `X3`. If this argument is a quoted name of one of the predictors, the added-variable plot is drawn for that predictor only.

`layout`	If set to a value like `c(1, 1)` or `c(4, 3)`, the layout of the graph will have this many rows and columns. If not set, the program will select an appropriate layout. If the number of graphs exceed nine, you must select the layout yourself, or you will get a maximum of nine per page. If `layout=NA`, the function does not set the layout and the user can use the `par` function to control the layout, for example to have plots from two models in the same graphics window.
`ask`	if `TRUE`, a menu is provided in the R Console for the user to select the term(s) to plot.
`xlab, ylab`	axis labels; if missing, labels will be supplied.
`main`	title for plot; if missing, a title will be supplied.
`...`	arguments passed down to method functions.
`term.name`	Quoted name of term in the model to be plotted; this argument is omitted for `leveragePlots`.
`id.method,labels,id.n,id.cex,id.col`	Arguments for the labelling of points. The default is `id.n=0` for labeling no points. See `showLabels` for details of these arguments.
`col`	color(s) of points
`col.lines`	color of the fitted line
`lwd`	line width; default is `2` (see `par`).
`grid`	If TRUE, the default, a light-gray background grid is put on the graph

Details

The function intended for direct use is `leveragePlots`.

The model can contain factors and interactions. A leverage plot can be drawn for each term in the model, including the constant.

`leveragePlot.glm` is a dummy function, which generates an error message.

Value

`NULL`. These functions are used for their side effect: producing plots.

Author(s)

John Fox `<jfox@mcmaster.ca>`

References

Cook, R. D. and Weisberg, S. (1991). Added Variable Plots in Linear Regression. In Stahel, W. and Weisberg, S. (eds.), *Directions in Robust Statistics and Diagnostics*. Springer, 47-60.

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Sall, J. (1990) Leverage plots for general linear hypotheses. *American Statistician* **44**, 308–315.

See Also

avPlots

Examples

```
leveragePlots(lm(prestige~(income+education)*type, data=Duncan))
```

---

linearHypothesis *Test Linear Hypothesis*

Description

Generic function for testing a linear hypothesis, and methods for linear models, generalized linear models, multivariate linear models, linear and generalized linear mixed-effects models, and other models that have methods for `coef` and `vcov`. For mixed-effects models, the tests are Wald chi-square tests for the fixed effects.

Usage

```
linearHypothesis(model, ...)

lht(model, ...)

## Default S3 method:
linearHypothesis(model, hypothesis.matrix, rhs=NULL,
test=c("Chisq", "F"), vcov.=NULL, singular.ok=FALSE, verbose=FALSE,
    coef. = coef(model), ...)

## S3 method for class 'lm'
linearHypothesis(model, hypothesis.matrix, rhs=NULL,
    test=c("F", "Chisq"), vcov.=NULL,
white.adjust=c(FALSE, TRUE, "hc3", "hc0", "hc1", "hc2", "hc4"),
singular.ok=FALSE, ...)

## S3 method for class 'glm'
linearHypothesis(model,  ...)

## S3 method for class 'nlsList'
linearHypothesis(model,  ..., vcov., coef.)

## S3 method for class 'mlm'
linearHypothesis(model, hypothesis.matrix, rhs=NULL, SSPE, V,
    test, idata, icontrasts=c("contr.sum", "contr.poly"), idesign, iterms,
    check.imatrix=TRUE, P=NULL, title="", singular.ok=FALSE, verbose=FALSE, ...)

## S3 method for class 'polr'
linearHypothesis(model, hypothesis.matrix, rhs=NULL, vcov.,
```

```
verbose=FALSE, ...)

## S3 method for class 'linearHypothesis.mlm'
print(x, SSP=TRUE, SSPE=SSP,
    digits=getOption("digits"), ...)

## S3 method for class 'lme'
linearHypothesis(model, hypothesis.matrix, rhs=NULL,
vcov.=NULL, singular.ok=FALSE, verbose=FALSE, ...)

## S3 method for class 'mer'
linearHypothesis(model, hypothesis.matrix, rhs=NULL,
vcov.=NULL, test=c("Chisq", "F"), singular.ok=FALSE, verbose=FALSE, ...)

## S3 method for class 'merMod'
linearHypothesis(model, hypothesis.matrix, rhs=NULL,
     vcov.=NULL, test=c("Chisq", "F"), singular.ok=FALSE, verbose=FALSE, ...)

## S3 method for class 'svyglm'
linearHypothesis(model, ...)



matchCoefs(model, pattern, ...)

## Default S3 method:
matchCoefs(model, pattern, coef.=coef, ...)

## S3 method for class 'lme'
matchCoefs(model, pattern, ...)

## S3 method for class 'mer'
matchCoefs(model, pattern, ...)

## S3 method for class 'merMod'
matchCoefs(model, pattern, ...)

## S3 method for class 'mlm'
matchCoefs(model, pattern, ...)
```

Arguments

`model` fitted model object. The default method of `linearHypothesis` works for models for which the estimated parameters can be retrieved by `coef` and the corresponding estimated covariance matrix by `vcov`. See the *Details* for more information.

`hypothesis.matrix` matrix (or vector) giving linear combinations of coefficients by rows, or a character vector giving the hypothesis in symbolic form (see *Details*).

`rhs` right-hand-side vector for hypothesis, with as many entries as rows in the hypothesis matrix; can be omitted, in which case it defaults to a vector of zeroes. For a multivariate linear model, `rhs` is a matrix, defaulting to 0.

`singular.ok` if `FALSE` (the default), a model with aliased coefficients produces an error; if `TRUE`, the aliased coefficients are ignored, and the hypothesis matrix should not have columns for them. For a multivariate linear model: will return the hypothesis and error SSP matrices even if the latter is singular; useful for computing univariate repeated-measures ANOVAs where there are fewer subjects than df for within-subject effects.

`idata` an optional data frame giving a factor or factors defining the intra-subject model for multivariate repeated-measures data. See *Details* for an explanation of the intra-subject design and for further explanation of the other arguments relating to intra-subject factors.

`icontrasts` names of contrast-generating functions to be applied by default to factors and ordered factors, respectively, in the within-subject "data"; the contrasts must produce an intra-subject model matrix in which different terms are orthogonal.

`idesign` a one-sided model formula using the "data" in `idata` and specifying the intra-subject design.

`iterms` the quoted name of a term, or a vector of quoted names of terms, in the intra-subject design to be tested.

`check.imatrix` check that columns of the intra-subject model matrix for different terms are mutually orthogonal (default, `TRUE`). Set to `FALSE` only if you have *already* checked that the intra-subject model matrix is block-orthogonal.

`P` transformation matrix to be applied to the repeated measures in multivariate repeated-measures data; if `NULL` *and* no intra-subject model is specified, no response-transformation is applied; if an intra-subject model is specified via the `idata`, `idesign`, and (optionally) `icontrasts` arguments, then `P` is generated automatically from the `iterms` argument.

`SSPE` in `linearHypothesis` method for `mlm` objects: optional error sum-of-squares-and-products matrix; if missing, it is computed from the model. In `print` method for `linearHypothesis.mlm` objects: if `TRUE`, print the sum-of-squares and cross-products matrix for error.

`test` character string, `"F"` or `"Chisq"`, specifying whether to compute the finite-sample F statistic (with approximate F distribution) or the large-sample Chi-squared statistic (with asymptotic Chi-squared distribution). For a multivariate linear model, the multivariate test statistic to report — one or more of `"Pillai"`, `"Wilks"`, `"Hotelling-Lawley"`, or `"Roy"`, with `"Pillai"` as the default.

`title` an optional character string to label the output.

`V` inverse of sum of squares and products of the model matrix; if missing it is computed from the model.

`vcov.` a function for estimating the covariance matrix of the regression coefficients, e.g., `hccm`, or an estimated covariance matrix for `model`. See also `white.adjust`.

`coef.` a vector of coefficient estimates. The default is to get the coefficient estimates from the `model` argument, but the user can input any vector of the correct length.

white.adjust	logical or character. Convenience interface to hccm (instead of using the argument vcov.). Can be set either to a character value specifying the type argument of hccm or TRUE, in which case "hc3" is used implicitly. The default is FALSE.
verbose	If TRUE, the hypothesis matrix, right-hand-side vector (or matrix), and estimated value of the hypothesis are printed to standard output; if FALSE (the default), the hypothesis is only printed in symbolic form and the value of the hypothesis is not printed.
x	an object produced by linearHypothesis.mlm.
SSP	if TRUE (the default), print the sum-of-squares and cross-products matrix for the hypothesis and the response-transformation matrix.
digits	minimum number of signficiant digits to print.
pattern	a regular expression to be matched against coefficient names.
...	arguments to pass down.

Details

linearHypothesis computes either a finite-sample F statistic or asymptotic Chi-squared statistic for carrying out a Wald-test-based comparison between a model and a linearly restricted model. The default method will work with any model object for which the coefficient vector can be retrieved by coef and the coefficient-covariance matrix by vcov (otherwise the argument vcov. has to be set explicitly). For computing the F statistic (but not the Chi-squared statistic) a df.residual method needs to be available. If a formula method exists, it is used for pretty printing.

The method for "lm" objects calls the default method, but it changes the default test to "F", supports the convenience argument white.adjust (for backwards compatibility), and enhances the output by the residual sums of squares. For "glm" objects just the default method is called (bypassing the "lm" method). The svyglm method also calls the default method.

The function lht also dispatches to linearHypothesis.

The hypothesis matrix can be supplied as a numeric matrix (or vector), the rows of which specify linear combinations of the model coefficients, which are tested equal to the corresponding entries in the right-hand-side vector, which defaults to a vector of zeroes.

Alternatively, the hypothesis can be specified symbolically as a character vector with one or more elements, each of which gives either a linear combination of coefficients, or a linear equation in the coefficients (i.e., with both a left and right side separated by an equals sign). Components of a linear expression or linear equation can consist of numeric constants, or numeric constants multiplying coefficient names (in which case the number precedes the coefficient, and may be separated from it by spaces or an asterisk); constants of 1 or -1 may be omitted. Spaces are always optional. Components are separated by plus or minus signs. Newlines or tabs in hypotheses will be treated as spaces. See the examples below.

If the user sets the arguments coef. and vcov., then the computations are done without reference to the model argument. This is like assuming that coef. is normally distibuted with estimated variance vcov. and the linearHypothesis will compute tests on the mean vector for coef., without actually using the model argument.

A linear hypothesis for a multivariate linear model (i.e., an object of class "mlm") can optionally include an intra-subject transformation matrix for a repeated-measures design. If the intra-subject transformation is absent (the default), the multivariate test concerns all of the corresponding coefficients for the response variables. There are two ways to specify the transformation matrix for the repeated measures:

1. The transformation matrix can be specified directly via the `P` argument.
2. A data frame can be provided defining the repeated-measures factor or factors via `idata`, with default contrasts given by the `icontrasts` argument. An intra-subject model-matrix is generated from the one-sided formula specified by the `idesign` argument; columns of the model matrix corresponding to different terms in the intra-subject model must be orthogonal (as is insured by the default contrasts). Note that the contrasts given in `icontrasts` can be overridden by assigning specific contrasts to the factors in `idata`. The repeated-measures transformation matrix consists of the columns of the intra-subject model matrix corresponding to the term or terms in `iterms`. In most instances, this will be the simpler approach, and indeed, most tests of interests can be generated automatically via the `Anova` function.

`matchCoefs` is a convenience function that can sometimes help in formulating hypotheses; for example `matchCoefs(mod, ":")` will return the names of all interaction coefficients in the model `mod`.

Value

For a univariate model, an object of class `"anova"` which contains the residual degrees of freedom in the model, the difference in degrees of freedom, Wald statistic (either `"F"` or `"Chisq"`), and corresponding p value.

For a multivariate linear model, an object of class `"linearHypothesis.mlm"`, which contains sums-of-squares-and-product matrices for the hypothesis and for error, degrees of freedom for the hypothesis and error, and some other information.

The returned object normally would be printed.

Author(s)

Achim Zeileis and John Fox `<jfox@mcmaster.ca>`

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Hand, D. J., and Taylor, C. C. (1987) *Multivariate Analysis of Variance and Repeated Measures: A Practical Approach for Behavioural Scientists.* Chapman and Hall.

O'Brien, R. G., and Kaiser, M. K. (1985) MANOVA method for analyzing repeated measures designs: An extensive primer. *Psychological Bulletin* **97**, 316–333.

See Also

`anova`, `Anova`, `waldtest`, `hccm`, `vcovHC`, `vcovHAC`, `coef`, `vcov`

Examples

```
mod.davis <- lm(weight ~ repwt, data=Davis)

## the following are equivalent:
linearHypothesis(mod.davis, diag(2), c(0,1))
linearHypothesis(mod.davis, c("(Intercept) = 0", "repwt = 1"))
linearHypothesis(mod.davis, c("(Intercept)", "repwt"), c(0,1))
linearHypothesis(mod.davis, c("(Intercept)", "repwt = 1"))

## use asymptotic Chi-squared statistic
linearHypothesis(mod.davis, c("(Intercept) = 0", "repwt = 1"), test = "Chisq")


## the following are equivalent:
  ## use HC3 standard errors via white.adjust option
linearHypothesis(mod.davis, c("(Intercept) = 0", "repwt = 1"),
    white.adjust = TRUE)
  ## covariance matrix *function*
linearHypothesis(mod.davis, c("(Intercept) = 0", "repwt = 1"), vcov = hccm)
  ## covariance matrix *estimate*
linearHypothesis(mod.davis, c("(Intercept) = 0", "repwt = 1"),
    vcov = hccm(mod.davis, type = "hc3"))

mod.duncan <- lm(prestige ~ income + education, data=Duncan)

## the following are all equivalent:
linearHypothesis(mod.duncan, "1*income - 1*education = 0")
linearHypothesis(mod.duncan, "income = education")
linearHypothesis(mod.duncan, "income - education")
linearHypothesis(mod.duncan, "1income - 1education = 0")
linearHypothesis(mod.duncan, "0 = 1*income - 1*education")
linearHypothesis(mod.duncan, "income-education=0")
linearHypothesis(mod.duncan, "1*income - 1*education + 1 = 1")
linearHypothesis(mod.duncan, "2income = 2*education")

mod.duncan.2 <- lm(prestige ~ type*(income + education), data=Duncan)
coefs <- names(coef(mod.duncan.2))

## test against the null model (i.e., only the intercept is not set to 0)
linearHypothesis(mod.duncan.2, coefs[-1])

## test all interaction coefficients equal to 0
linearHypothesis(mod.duncan.2, coefs[grep(":", coefs)], verbose=TRUE)
linearHypothesis(mod.duncan.2, matchCoefs(mod.duncan.2, ":"), verbose=TRUE) # equivalent

## a multivariate linear model for repeated-measures data
## see ?OBrienKaiser for a description of the data set used in this example.

mod.ok <- lm(cbind(pre.1, pre.2, pre.3, pre.4, pre.5,
                     post.1, post.2, post.3, post.4, post.5,
                     fup.1, fup.2, fup.3, fup.4, fup.5) ~  treatment*gender,
                data=OBrienKaiser)
```

```
coef(mod.ok)

## specify the model for the repeated measures:
phase <- factor(rep(c("pretest", "posttest", "followup"), c(5, 5, 5)),
    levels=c("pretest", "posttest", "followup"))
hour <- ordered(rep(1:5, 3))
idata <- data.frame(phase, hour)
idata

## test the four-way interaction among the between-subject factors
## treatment and gender, and the intra-subject factors
## phase and hour

linearHypothesis(mod.ok, c("treatment1:gender1", "treatment2:gender1"),
    title="treatment:gender:phase:hour", idata=idata, idesign=~phase*hour,
    iterms="phase:hour")

## mixed-effects models examples:

## Not run:
library(nlme)
example(lme)
linearHypothesis(fm2, "age = 0")

## End(Not run)

## Not run:
library(lme4)
example(glmer)
linearHypothesis(gm1, matchCoefs(gm1, "period"))

## End(Not run)
```

`logit` *Logit Transformation*

Description

Compute the logit transformation of proportions or percentages.

Usage

```
logit(p, percents=range.p[2] > 1, adjust)
```

Arguments

`p`	numeric vector or array of proportions or percentages.
`percents`	`TRUE` for percentages.
`adjust`	adjustment factor to avoid proportions of 0 or 1; defaults to `0` if there are no such proportions in the data, and to `.025` if there are.

Details

Computes the logit transformation logit $= \log[p/(1-p)]$ for the proportion p.

If $p = 0$ or 1, then the logit is undefined. `logit` can remap the proportions to the interval `(adjust, 1 - adjust)` prior to the transformation. If it adjusts the data automatically, `logit` will print a warning message.

Value

a numeric vector or array of the same shape and size as p.

Author(s)

John Fox <jfox@mcmaster.ca>

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

See Also

`probabilityAxis`

Examples

```
options(digits=4)
logit(.1*0:10)
##  [1] -3.6636 -1.9924 -1.2950 -0.8001 -0.3847  0.0000  0.3847
##  [8]  0.8001  1.2950  1.9924  3.6636
## Warning message:
## proportions remapped to (0.025, 0.975) in: logit(0.1 * 0:10)

logit(.1*0:10, adjust=0)
##  [1]    -Inf -2.1972 -1.3863 -0.8473 -0.4055  0.0000  0.4055
##  [8]  0.8473  1.3863  2.1972     Inf
```

Mandel *Contrived Collinear Data*

Description

The `Mandel` data frame has 8 rows and 3 columns.

Usage

```
Mandel
```

Format

This data frame contains the following columns:

x1 first predictor.

x2 second predictor.

y response.

Source

Mandel, J. (1982) Use of the singular value decomposition in regression analysis. *The American Statistician* **36**, 15–24.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

`Migration` *Canadian Interprovincial Migration Data*

Description

The `Migration` data frame has 90 rows and 8 columns.

Usage

```
Migration
```

Format

This data frame contains the following columns:

source Province of origin (source). A factor with levels: `ALTA`, Alberta; `BC`, British Columbia; `MAN`, Manitoba; `NB`, New Brunswick; `NFLD`, New Foundland; `NS`, Nova Scotia; `ONT`, Ontario; `PEI`, Prince Edward Island; `QUE`, Quebec; `SASK`, Saskatchewan.

destination Province of destination (1971 residence). A factor with levels: `ALTA`, Alberta; `BC`, British Columbia; `MAN`, Manitoba; `NB`, New Brunswick; `NFLD`, New Foundland; `NS`, Nova Scotia; `ONT`, Ontario; `PEI`, Prince Edward Island; `QUE`, Quebec; `SASK`, Saskatchewan.

migrants Number of migrants (from source to destination) in the period 1966–1971.

distance Distance (between principal cities of provinces): NFLD, St. John; PEI, Charlottetown; NS, Halifax; NB, Fredricton; QUE, Montreal; ONT, Toronto; MAN, Winnipeg; SASK, Regina; ALTA, Edmonton; BC, Vancouver.

pops66 1966 population of source province.

pops71 1971 population of source province.

popd66 1966 population of destination province.

popd71 1971 population of destination province.

Details

There is one record in the data file for each migration stream. You can average the 1966 and 1971 population figures for each of the source and destination provinces.

Source

Canada (1962) *Map*. Department of Mines and Technical Surveys.

Canada (1971) *Census of Canada*. Statistics Canada, Vol. 1, Part 2 [Table 32].

Canada (1972) *Canada Year Book*. Statistics Canada [p. 1369].

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

---

mmps *Marginal Model Plotting*

Description

For a regression object, plots the response on the vertical axis versus a linear combination u of terms in the mean function on the horizontal axis. Added to the plot are a `loess` smooth for the graph, along with a loess smooth from the plot of the fitted values on u. `mmps` is an alias for `marginalModelPlots`, and `mmp` is an alias for `marginalModelPlot`.

Usage

```
marginalModelPlots(...)

mmps(model, terms= ~ ., fitted=TRUE, layout=NULL, ask,
        main, groups, key=TRUE, ...)

marginalModelPlot(...)

## S3 method for class 'lm'
mmp(model, variable, sd = FALSE,
    xlab = deparse(substitute(variable)),
    smoother = loessLine, smoother.args=list(span=2/3),
    key=TRUE, pch, groups=NULL, ...)

## Default S3 method:
mmp(model, variable, sd = FALSE,
    xlab = deparse(substitute(variable)), smoother=loessLine,
    smoother.args, key=TRUE, pch, groups=NULL,
    col.line = palette()[c(4, 2)], col=palette()[1],
    labels, id.method="y",
    id.n=if(id.method[1]=="identify") Inf else 0,
```

```
    id.cex=1, id.col=palette()[1], grid=TRUE, ...)

## S3 method for class 'glm'
mmp(model, variable, sd = FALSE,
    xlab = deparse(substitute(variable)), smoother=gamLine,
    smoother.args=list(k=3), key=TRUE, pch, groups=NULL,
    col.line = palette()[c(4, 2)], col=palette()[1],
    labels, id.method="y",
    id.n=if(id.method[1]=="identify") Inf else 0,
    id.cex=1, id.col=palette()[1], grid=TRUE, ...)
```

Arguments

model	A regression object, usually of class either `lm` or `glm`, for which there is a `predict` method defined.
terms	A one-sided formula. A marginal model plot will be drawn for each variable on the right-side of this formula that is not a factor. The default is `~ .`, which specifies that all the terms in `formula(object)` will be used. If a conditioning argument is given, eg `terms = ~. | a`, then separate colors and smoothers are used for each unique non-missing value of `a`. See examples below.
fitted	If the default `TRUE`, then a marginal model plot in the direction of the fitted values or linear predictor of a generalized linear model will be drawn.
layout	If set to a value like `c(1, 1)` or `c(4, 3)`, the layout of the graph will have this many rows and columns. If not set, the program will select an appropriate layout. If the number of graphs exceed nine, you must select the layout yourself, or you will get a maximum of nine per page. If `layout=NA`, the function does not set the layout and the user can use the `par` function to control the layout, for example to have plots from two models in the same graphics window.
ask	If `TRUE`, ask before clearing the graph window to draw more plots.
main	Main title for the array of plots. Use `main=""` to suppress the title; if missing, a title will be supplied.
...	Additional arguments passed from `mmps` to `mmp` and then to `plot`. Users should generally use `mmps`, or equivalently `marginalModelPlots`.
variable	The quantity to be plotted on the horizontal axis. The default is the predicted values `predict(object)`. Can be any other vector of length equal to the number of observations in the object. Thus the `mmp` function can be used to get a marginal model plot versus any predictor or term while the `mmps` function can be used only to get marginal model plots for the first-order terms in the formula. In particular, terms defined by a spline basis are skipped by `mmps`, but you can use `mmp` to get the plot for the variable used to define the splines.
sd	If `TRUE`, compare sd smooths. For a binomial regression with all sample sizes equal to one, this argument is ignored as the SD bounds don't make any sense.
xlab	label for horizontal axis
smoother	the name of the smoother to use, selected from the choices descripbed at `ScatterplotSmoothers`. For linear models and the default method, the default smoother is the function is the function `loessLine`. For generalized linear models the default is `gamLine`, using the `gam` package and using splines.

smoother.args
: arguments passed to the smoother. For linear models the defaults match the smoother used before September 2012, and may be changed later. See ScatterplotSmoothers.

groups
: The name of a vector that specifies a grouping variable for separate colors/smoothers. This can also be specified as a conditioning argument on the terms argument.

key
: If TRUE, include a key at the top of the plot, if FALSE omit the key. If grouping is present, the key is only printed for the upper-left plot.

id.method,labels,id.n,id.cex,id.col
: Arguments for labelling points. The default id.n=0 suppresses labelling, and setting this argument greater than zero will include labelling. See showLabels for these arguments.

pch
: plotting character to use if no grouping is present.

col.line
: colors for data and model smooth, respectively. Using the default palette, these are blue and red.

col
: color(s) for the plotted points.

grid
: If TRUE, the default, a light-gray background grid is put on the graph

Details

mmp and marginalModelPlot draw one marginal model plot against whatever is specified as the horizontal axis. mmps and marginalModelPlots draws marginal model plots versus each of the terms in the terms argument and versus fitted values. mmps skips factors and interactions if they are specified in the terms argument. Terms based on polynomials or on splines (or potentially any term that is represented by a matrix of predictors) will be used to form a marginal model plot by returning a linear combination of the terms. For example, if you specify terms ~ X1 + poly(X2, 3) and poly(X2, 3) was part of the original model formula, the horizontal axis of the marginal model plot will be the value of predict(model, type="terms")[, "poly(X2, 3)"]). If the predict method for the model you are using doesn't support type="terms", then the polynomial/spline term is skipped. Adding a conditioning variable, e.g., terms = ~ a + b | c, will produce marginal model plots for a and b with different colors and smoothers for each unique non-missing value of c.

The smoothers used were changed in September 2012. For linear models, the default smoother is still loess with the same smoothing parameters as were used in the past, but these can be changed with the argument smoother.args. For generalized linear models, the default smoother uses gamLine, fitting a generalized additive model with the same family, link and weights as the fit of the model. SD smooths are not computed for for generalized linear models.

For generalized linear models the default number of elements in the spline basis is k=3; this is done to allow fitting for predictors with just a few support points. If you have many support points you may wish to set k to a higher number, or k=-1 for the default used by gam.

Value

Used for its side effect of producing plots.

Author(s)

Sanford Weisberg, <sandy@umn.edu>

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition. Sage.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition, Wiley, Chapter 8.

See Also

ScatterplotSmoothers, plot

Examples

```
## Not run:
c1 <- lm(infant.mortality ~ gdp, UN)
mmps(c1)
c2 <- update(c1, ~ poly(gdp, 4), data=na.omit(UN))
# plot against predict(c2, type="terms")[, "poly(gdp, 4)"] and
# and against gdp
mmps(c2, ~ poly(gdp,4) + gdp)
# include SD lines
p1 <- lm(prestige ~ income + education, Prestige)
mmps(p1, sd=TRUE)
# logisitic regression example
# smoothers return warning messages.
mmps(p1, ~. | type)
# fit a separate smoother and color for each type of occupation.

m1 <- glm(lfp ~ ., family=binomial, data=Mroz)
mmps(m1)

## End(Not run)
```

Moore *Status, Authoritarianism, and Conformity*

Description

The Moore data frame has 45 rows and 4 columns. The data are for subjects in a social-psychological experiment, who were faced with manipulated disagreement from a partner of either of low or high status. The subjects could either conform to the partner's judgment or stick with their own judgment.

Usage

```
Moore
```

Format

This data frame contains the following columns:

partner.status Partner's status. A factor with levels: high, low.

conformity Number of conforming responses in 40 critical trials.

fcategory F-Scale Categorized. A factor with levels (note levels out of order): `high`, `low`, `medium`.

fscore Authoritarianism: F-Scale score.

Source

Moore, J. C., Jr. and Krupat, E. (1971) Relationship between source status, authoritarianism and conformity in a social setting. *Sociometry* **34**, 122–134.

Personal communication from J. Moore, Department of Sociology, York University.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

`Mroz` *U.S. Women's Labor-Force Participation*

Description

The `Mroz` data frame has 753 rows and 8 columns. The observations, from the Panel Study of Income Dynamics (PSID), are married women.

Usage

```
Mroz
```

Format

This data frame contains the following columns:

lfp labor-force participation; a factor with levels: `no`; `yes`.

k5 number of children 5 years old or younger.

k618 number of children 6 to 18 years old.

age in years.

wc wife's college attendance; a factor with levels: `no`; `yes`.

hc husband's college attendance; a factor with levels: `no`; `yes`.

lwg log expected wage rate; for women in the labor force, the actual wage rate; for women not in the labor force, an imputed value based on the regression of `lwg` on the other variables.

inc family income exclusive of wife's income.

Source

Mroz, T. A. (1987) The sensitivity of an empirical model of married women's hours of work to economic and statistical assumptions. *Econometrica* **55**, 765–799.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. (2000) *Multiple and Generalized Nonparametric Regression.* Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Long. J. S. (1997) *Regression Models for Categorical and Limited Dependent Variables.* Sage.

`ncvTest` *Score Test for Non-Constant Error Variance*

Description

Computes a score test of the hypothesis of constant error variance against the alternative that the error variance changes with the level of the response (fitted values), or with a linear combination of predictors.

Usage

```
ncvTest(model, ...)

## S3 method for class 'lm'
ncvTest(model, var.formula, ...)

## S3 method for class 'glm'
ncvTest(model, ...) # to report an error
```

Arguments

`model`	a weighted or unweighted linear model, produced by `lm`.
`var.formula`	a one-sided formula for the error variance; if omitted, the error variance depends on the fitted values.
`...`	arguments passed down to methods functions; not currently used.

Details

This test is often called the Breusch-Pagan test; it was independently suggested with some extension by Cook and Weisberg (1983).

`ncvTest.glm` is a dummy function to generate an error when a `glm` model is used.

Value

The function returns a `chisqTest` object, which is usually just printed.

Author(s)

John Fox <`jfox@mcmaster.ca`>, Sandy Weisberg <`sandy@umn.edu`>

References

Breusch, T. S. and Pagan, A. R. (1979) A simple test for heteroscedasticity and random coefficient variation. *Econometrica* **47**, 1287–1294.

Cook, R. D. and Weisberg, S. (1983) Diagnostics for heteroscedasticity in regression. *Biometrika* **70**, 1–10.

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition, Wiley.

See Also

`hccm`, `spreadLevelPlot`

Examples

```
ncvTest(lm(interlocks ~ assets + sector + nation, data=Ornstein))

ncvTest(lm(interlocks ~ assets + sector + nation, data=Ornstein),
    ~ assets + sector + nation, data=Ornstein)
```

OBrienKaiser — *O'Brien and Kaiser's Repeated-Measures Data*

Description

These contrived repeated-measures data are taken from O'Brien and Kaiser (1985). The data are from an imaginary study in which 16 female and male subjects, who are divided into three treatments, are measured at a pretest, postest, and a follow-up session; during each session, they are measured at five occasions at intervals of one hour. The design, therefore, has two between-subject and two within-subject factors.

The contrasts for the `treatment` factor are set to $-2, 1, 1$ and $0, -1, 1$. The contrasts for the `gender` factor are set to `contr.sum`.

Usage

```
OBrienKaiser
```

Format

A data frame with 16 observations on the following 17 variables.

`treatment` a factor with levels `control` `A` `B`

`gender` a factor with levels `F` `M`

`pre.1` pretest, hour 1

`pre.2` pretest, hour 2

`pre.3` pretest, hour 3

`pre.4` pretest, hour 4

`pre.5` pretest, hour 5

`post.1` posttest, hour 1

`post.2` posttest, hour 2

`post.3` posttest, hour 3

`post.4` posttest, hour 4

`post.5` posttest, hour 5

`fup.1` follow-up, hour 1

`fup.2` follow-up, hour 2

`fup.3` follow-up, hour 3

`fup.4` follow-up, hour 4

`fup.5` follow-up, hour 5

Source

O'Brien, R. G., and Kaiser, M. K. (1985) MANOVA method for analyzing repeated measures designs: An extensive primer. *Psychological Bulletin* **97**, 316–333, Table 7.

Examples

```
OBrienKaiser
contrasts(OBrienKaiser$treatment)
contrasts(OBrienKaiser$gender)
```

`Ornstein` *Interlocking Directorates Among Major Canadian Firms*

Description

The `Ornstein` data frame has 248 rows and 4 columns. The observations are the 248 largest Canadian firms with publicly available information in the mid-1970s. The names of the firms were not available.

Usage

```
Ornstein
```

Format

This data frame contains the following columns:

assets Assets in millions of dollars.

sector Industrial sector. A factor with levels: `AGR`, agriculture, food, light industry; `BNK`, banking; `CON`, construction; `FIN`, other financial; `HLD`, holding companies; `MAN`, heavy manufacturing; `MER`, merchandizing; `MIN`, mining, metals, etc.; `TRN`, transport; `WOD`, wood and paper.

nation Nation of control. A factor with levels: `CAN`, Canada; `OTH`, other foreign; `UK`, Britain; `US`, United States.

interlocks Number of interlocking director and executive positions shared with other major firms.

Source

Ornstein, M. (1976) The boards and executives of the largest Canadian corporations. *Canadian Journal of Sociology* **1**, 411–437.

Personal communication from M. Ornstein, Department of Sociology, York University.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

`outlierTest` *Bonferroni Outlier Test*

Description

Reports the Bonferroni p-values for Studentized residuals in linear and generalized linear models, based on a t-test for linear models and normal-distribution test for generalized linear models.

Usage

```
outlierTest(model, ...)

## S3 method for class 'lm'
outlierTest(model, cutoff=0.05, n.max=10, order=TRUE,
labels=names(rstudent), ...)

## S3 method for class 'outlierTest'
print(x, digits=5, ...)
```

Arguments

model	an lm or glm model object.
cutoff	observations with Bonferonni p-values exceeding cutoff are not reported, unless no observations are nominated, in which case the one with the largest Studentized residual is reported.
n.max	maximum number of observations to report (default, 10).
order	report Studenized residuals in descending order of magnitude? (default, TRUE).
labels	an optional vector of observation names.
...	arguments passed down to methods functions.
x	outlierTest object.
digits	number of digits for reported p-values.

Details

For a linear model, p-values reported use the t distribution with degrees of freedom one less than the residual df for the model. For a generalized linear model, p-values are based on the standard-normal distribution. The Bonferroni adjustment multiplies the usual two-sided p-value by the number of observations. The lm method works for glm objects. To show all of the observations set cutoff=Inf and n.max=Inf.

Value

an object of class outlierTest, which is normally just printed.

Author(s)

John Fox <jfox@mcmaster.ca> and Sanford Weisberg

References

Cook, R. D. and Weisberg, S. (1982) *Residuals and Influence in Regression.* Chapman and Hall.

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition, Wiley.

Williams, D. A. (1987) Generalized linear model diagnostics using the deviance and single case deletions. *Applied Statistics* **36**, 181–191.

Examples

```
outlierTest(lm(prestige ~ income + education, data=Duncan))
```

panel.car *Panel Function for Coplots*

Description

a panel function for use with `coplot` that plots points, a lowess line, and a regression line.

Usage

```
panel.car(x, y, col, pch, cex=1, span=0.5, lwd=2,
  reg.line=lm, lowess.line=TRUE, ...)
```

Arguments

`x`	vector giving horizontal coordinates.
`y`	vector giving vertical coordinates.
`col`	point color.
`pch`	plotting character for points.
`cex`	character expansion factor for points.
`span`	span for lowess smoother.
`lwd`	line width, default is `2`.
`reg.line`	function to compute coefficients of regression line, or `FALSE` for no line.
`lowess.line`	if `TRUE` plot lowess smooth.
`...`	other arguments to pass to functions `lines` and `regLine`.

Value

`NULL`. This function is used for its side effect: producing a panel in a coplot.

Author(s)

John Fox <jfox@mcmaster.ca>

See Also

`coplot`, `regLine`

Examples

```
coplot(prestige ~ income|education, panel=panel.car,
  col="red", data=Prestige)
```

`plot.powerTransform` *plot Method for powerTransform Objects*

Description

This function provides a simple function for plotting data using power transformations.

Usage

```
## S3 method for class 'powerTransform'
plot(x, z = NULL, round = TRUE, plot = pairs, ...)
```

Arguments

`x`	name of the power transformation object
`z`	Additional variables of the same length as those used to get the transformation to be plotted, default is `NULL`.
`round`	If `TRUE`, the default, use rounded transforms, if `FALSE` use the MLEs.
`plot`	Plotting method. Default is `pairs`. Another possible choice is `scatterplot.matrix` from the `car` package.
`...`	Optional arguments passed to the plotting method

Details

The data used to estimate transformations using `powerTransform` are plotted in the transformed scale.

Value

None. Produces a graph as a side-effect.

Author(s)

Sanford Weisberg, <sandy@umn.edu>

References

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition. Wiley.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Linear Regression*, Second Edition, Sage.

See Also

`powerTransform`

Examples

```
summary(a3 <- powerTransform(cbind(len, ADT, trks, sigs1) ~ hwy, Highway1))
with(Highway1, plot(a3, z=rate, col=as.numeric(hwy)))
```

Pottery *Chemical Composition of Pottery*

Description

The data give the chemical composition of ancient pottery found at four sites in Great Britain. They appear in Hand, et al. (1994), and are used to illustrate MANOVA in the SAS Manual. (Suggested by Michael Friendly.)

Usage

```
Pottery
```

Format

A data frame with 26 observations on the following 6 variables.

`Site` a factor with levels `AshleyRails` `Caldicot` `IsleThorns` `Llanedyrn`

`Al` Aluminum

`Fe` Iron

`Mg` Magnesium

`Ca` Calcium

`Na` Sodium

Source

Hand, D. J., Daly, F., Lunn, A. D., McConway, K. J., and E., O. (1994) *A Handbook of Small Data Sets*. Chapman and Hall.

Examples

```
Pottery
```

powerTransform *Finding Univariate or Multivariate Power Transformations*

Description

powerTransform computes members of families of transformations indexed by one parameter, the Box-Cox power family, or the Yeo and Johnson (2000) family, or the basic power family, interpreting zero power as logarithmic. The family can be modified to have Jacobian one, or not, except for the basic power family.

Usage

```
powerTransform(object,...)

## Default S3 method:
powerTransform(object,...)

## S3 method for class 'lm'
powerTransform(object, ...)

## S3 method for class 'formula'
powerTransform(object, data, subset, weights, na.action,
  ...)
```

Arguments

object	This can either be an object of class lm, a formula, or a matrix or vector; see below.
data	A data frame or environment, as in lm.
subset	Case indices to be used, as in lm.
weights	Weights as in lm.
na.action	Missing value action, as in 'lm'.
...	Additional arguments that are passed to estimateTransform, which does the actual computing, or the optim function, which does the maximization. See the documentation for these functions for the arguments that are permitted, including family for setting the power transformation family.

Details

The function powerTransform is used to estimate normalizing transformations of a univariate or a multivariate random variable. For a univariate transformation, a formula like z~x1+x2+x3 will find estimate a transformation for the response z from the family of transformations indexed by the parameter lambda that makes the residuals from the regression of the transformed z on the predictors as closed to normally distributed as possible. This generalizes the Box and Cox (1964) transformations to normality only by allowing for families other than the power transformations used in that paper.

For a formula like `cbind(y1,y2,y3)~x1+x2+x3`, the three variables on the left-side are all transformed, generally with different transformations to make all the residuals as close to normally distributed as possible. `cbind(y1,y2,y3)~1` would specify transformations to multivariate normality with no predictors. This generalizes the multivariate power transformations suggested by Velilla (1993) by allowing for different families of transformations, and by allowing for predictors. Cook and Weisberg (1999) and Weisberg (2005) suggest the usefulness of transforming a set of predictors `z1, z2, z3` for multivariate normality and for transforming for multivariate normality conditional on levels of a factor, which is equivalent to setting the predictors to be indicator variables for that factor.

Specifying the first argument as a vector, for example `powerTransform(ais$LBM)`, is equivalent to `powerTransform(LBM ~ 1, ais)`. Similarly, `powerTransform( cbind(ais$LBM, ais$SSF))`, where the first argument is a matrix rather than a formula is equivalent to `powerTransform(cbind(LBM, SSF) ~ 1, ais)`.

Two families of power transformations are available. The bcPower family of *scaled power transformations*, `family="bctrans"`, equals $(U^{\lambda} - 1)/\lambda$ for $\lambda \neq 0$, and $\log(U)$ if $\lambda = 0$.

If `family="yjPower"` then the Yeo-Johnson transformations are used. This is is Box-Cox transformation of $U + 1$ for nonnegative values, and of $|U| + 1$ with parameter $2 - \lambda$ for U negative.

Other families can be added by writing a function whose first argument is a matrix or vector to be transformed, and whose second argument is the value of the transformation parameter. The function must return modified transformations so that the Jacobian of the transformation is equal to one; see Cook and Weisberg (1982).

The function `powerTransform` is a front-end for `estimateTransform`.

The function `testTransform` is used to obtain likelihood ratio tests for any specified value for the transformation parameters. It is used by the summary method for powerTransform objects.

Value

The result of `powerTransform` is an object of class `powerTransform` that gives the estimates of the the transformation parameters and related statistics. The `print` method for the object will display the estimates only; the `summary` method provides both the estimates, standard errors, marginal Wald confidence intervals and relevant likelihood ratio tests.

Several helper functions are available. The `coef` method returns the estimated transformation parameters, while `coef(object,round=TRUE)` will return the transformations rounded to nearby convenient values within 1.96 standard errors of the mle. The `vcov` function returns the estimated covariance matrix of the estimated transformation parameters. A `print` method is used to print the objects and `summary` to provide more information. By default the summary method calls `testTransform` and provides likelihood ratio type tests that all transformation parameters equal one and that all transformation parameters equal zero, for log transformations, and for a convenient rounded value not far from the mle. The function can be called directly to test any other value for λ.

Author(s)

Sanford Weisberg, <sandy@umn.edu>

References

Box, G. E. P. and Cox, D. R. (1964) An analysis of transformations. *Journal of the Royal Statisistical Society, Series B*. 26 211-46.

Cook, R. D. and Weisberg, S. (1999) *Applied Regression Including Computing and Graphics*. Wiley.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Velilla, S. (1993) A note on the multivariate Box-Cox transformation to normality. *Statistics and Probability Letters*, 17, 259-263.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition. Wiley.

Yeo, I. and Johnson, R. (2000) A new family of power transformations to improve normality or symmetry. *Biometrika*, 87, 954-959.

See Also

estimateTransform, testTransform, optim, bcPower, transform.

Examples

```
# Box Cox Method, univariate
summary(p1 <- powerTransform(cycles ~ len + amp + load, Wool))

# fit linear model with transformed response:
coef(p1, round=TRUE)
summary(m1 <- lm(bcPower(cycles, p1$roundlam) ~ len + amp + load, Wool))

# Multivariate Box Cox
summary(powerTransform(cbind(len, ADT, trks, sigs1) ~ 1, Highway1))

# Multivariate transformation to normality within levels of 'hwy'
summary(a3 <- powerTransform(cbind(len, ADT, trks, sigs1) ~ hwy, Highway1))

# test lambda = (0 0 0 -1)
testTransform(a3, c(0, 0, 0, -1))

# save the rounded transformed values, plot them with a separate
# color for each highway type
transformedY <- bcPower(with(Highway1, cbind(len, ADT, trks, sigs1)),
                coef(a3, round=TRUE))
## Not run: pairs(transformedY, col=as.numeric(Highway1$hwy))
```

Prestige — *Prestige of Canadian Occupations*

Description

The Prestige data frame has 102 rows and 6 columns. The observations are occupations.

Usage

```
Prestige
```

Format

This data frame contains the following columns:

education Average education of occupational incumbents, years, in 1971.

income Average income of incumbents, dollars, in 1971.

women Percentage of incumbents who are women.

prestige Pineo-Porter prestige score for occupation, from a social survey conducted in the mid-1960s.

census Canadian Census occupational code.

type Type of occupation. A factor with levels (note: out of order): `bc`, Blue Collar; `prof`, Professional, Managerial, and Technical; `wc`, White Collar.

Source

Canada (1971) *Census of Canada*. Vol. 3, Part 6. Statistics Canada [pp. 19-1–19-21].

Personal communication from B. Blishen, W. Carroll, and C. Moore, Departments of Sociology, York University and University of Victoria.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

`qqPlot` *Quantile-Comparison Plots*

Description

Plots empirical quantiles of a variable, or of studentized residuals from a linear model, against theoretical quantiles of a comparison distribution.

Usage

```
qqPlot(x, ...)

qqp(...)

## Default S3 method:
qqPlot(x, distribution="norm", ...,
ylab=deparse(substitute(x)), xlab=paste(distribution, "quantiles"),
main=NULL, las=par("las"),
```

```
envelope=.95,
col=palette()[1], col.lines=palette()[2], lwd=2, pch=1, cex=par("cex"),
line=c("quartiles", "robust", "none"),
labels = if(!is.null(names(x))) names(x) else seq(along=x),
id.method = "y",
id.n =if(id.method[1]=="identify") Inf else 0,
id.cex=1, id.col=palette()[1], grid=TRUE)

## S3 method for class 'lm'
qqPlot(x, xlab=paste(distribution, "Quantiles"),
ylab=paste("Studentized Residuals(", deparse(substitute(x)), ")",
sep=""), main=NULL,
distribution=c("t", "norm"), line=c("robust", "quartiles", "none"),
las=par("las"), simulate=TRUE, envelope=.95,
reps=100, col=palette()[1], col.lines=palette()[2], lwd=2,
pch=1, cex=par("cex"),
labels, id.method = "y",
id.n = if(id.method[1]=="identify") Inf else 0,
id.cex=1, id.col=palette()[1], grid=TRUE, ...)
```

Arguments

x	vector of numeric values or lm object.
distribution	root name of comparison distribution – e.g., "norm" for the normal distribution; t for the t-distribution.
ylab	label for vertical (empirical quantiles) axis.
xlab	label for horizontal (comparison quantiles) axis.
main	label for plot.
envelope	confidence level for point-wise confidence envelope, or FALSE for no envelope.
las	if 0, ticks labels are drawn parallel to the axis; set to 1 for horizontal labels (see par).
col	color for points; the default is the *first* entry in the current color palette (see palette and par).
col.lines	color for lines; the default is the *second* entry in the current color palette.
pch	plotting character for points; default is 1 (a circle, see par).
cex	factor for expanding the size of plotted symbols; the default is 1.
labels	vector of text strings to be used to identify points, defaults to names(x) or observation numbers if names(x) is NULL.
id.method	point identification method. The default id.method="y" will identify the id.n points with the largest value of abs(y-mean(y)). See showLabels for other options.
id.n	number of points labeled. If id.n=0, the default, no point identification.
id.cex	set size of the text for point labels; the default is cex (which is typically 1).
id.col	color for the point labels.

lwd	line width; default is 2 (see par).
line	"quartiles" to pass a line through the quartile-pairs, or "robust" for a robust-regression line; the latter uses the rlm function in the MASS package. Specifying line = "none" suppresses the line.
simulate	if TRUE calculate confidence envelope by parametric bootstrap; for lm object only. The method is due to Atkinson (1985).
reps	integer; number of bootstrap replications for confidence envelope.
...	arguments such as df to be passed to the appropriate quantile function.
grid	If TRUE, the default, a light-gray background grid is put on the graph

Details

Draws theoretical quantile-comparison plots for variables and for studentized residuals from a linear model. A comparison line is drawn on the plot either through the quartiles of the two distributions, or by robust regression.

Any distribution for which quantile and density functions exist in R (with prefixes q and d, respectively) may be used. Studentized residuals from linear models are plotted against the appropriate t-distribution.

The function qqp is an abbreviation for qqPlot.

Value

These functions return the labels of identified points.

Author(s)

John Fox <jfox@mcmaster.ca>

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Atkinson, A. C. (1985) *Plots, Transformations, and Regression.* Oxford.

See Also

qqplot, qqnorm, qqline, showLabels

Examples

```
x<-rchisq(100, df=2)
qqPlot(x)
qqPlot(x, dist="chisq", df=2)

qqPlot(lm(prestige ~ income + education + type, data=Duncan),
envelope=.99)
```

`Quartet` *Four Regression Datasets*

Description

The `Quartet` data frame has 11 rows and 5 columns. These are contrived data.

Usage

```
Quartet
```

Format

This data frame contains the following columns:

x X-values for datasets 1–3.

y1 Y-values for dataset 1.

y2 Y-values for dataset 2.

y3 Y-values for dataset 3.

x4 X-values for dataset 4.

y4 Y-values for dataset 4.

Source

Anscombe, F. J. (1973) Graphs in statistical analysis. *American Statistician* **27**, 17–21.

`recode` *Recode a Variable*

Description

Recodes a numeric vector, character vector, or factor according to simple recode specifications. `Recode` is an alias for `recode` that avoids name clashes with packages, such as **Hmisc**, that have a `recode` function.

Usage

```
recode(var, recodes, as.factor.result, as.numeric.result=TRUE, levels)

Recode(...)
```

Arguments

`var`	numeric vector, character vector, or factor.
`recodes`	character string of recode specifications: see below.
`as.factor.result`	return a factor; default is `TRUE` if `var` is a factor, `FALSE` otherwise.
`as.numeric.result`	if `TRUE` (the default), and `as.factor.result` is `FALSE`, then the result will be coerced to numeric if all values in the result are numerals—i.e., represent numbers.
`levels`	an optional argument specifying the order of the levels in the returned factor; the default is to use the sort order of the level names.
`...`	arguments to be passed to `recode`.

Details

Recode specifications appear in a character string, separated by semicolons (see the examples below), of the form `input=output`. If an input value satisfies more than one specification, then the first (from left to right) applies. If no specification is satisfied, then the input value is carried over to the result. `NA` is allowed on input and output. Several recode specifications are supported:

single value For example, `0=NA`.

vector of values For example, `c(7,8,9)='high'`.

range of values For example, `7:9='C'`. The special values `lo` and `hi` may appear in a range. For example, `lo:10=1`. *Note:* `:` is *not* the R sequence operator.

`else` everything that does not fit a previous specification. For example, `else=NA`. Note that `else` matches *all* otherwise unspecified values on input, including `NA`.

If all of the output values are numeric, and if `as.factor.result` is `FALSE`, then a numeric result is returned; if `var` is a factor, then by default so is the result.

Value

a recoded vector of the same length as `var`.

Author(s)

John Fox <jfox@mcmaster.ca>

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

See Also

`cut`, `factor`

Examples

```
x<-rep(1:3,3)
x
## [1] 1 2 3 1 2 3 1 2 3
recode(x, "c(1,2)='A';
else='B'")
## [1] "A" "A" "B" "A" "A" "B" "A" "A" "B"
Recode(x, "1:2='A'; 3='B'")
## [1] "A" "A" "B" "A" "A" "B" "A" "A" "B"
```

---

regLine *Plot Regression Line*

Description

Plots a regression line on a scatterplot; the line is plotted between the minimum and maximum x-values.

Usage

```
regLine(mod, col=palette()[2], lwd=2, lty=1,...)
```

Arguments

mod	a model, such as produced by `lm`, that responds to the `coef` function by returning a 2-element vector, whose elements are interpreted respectively as the intercept and slope of a regresison line.
col	color for points and lines; the default is the *second* entry in the current color palette (see `palette` and `par`).
lwd	line width; default is `2` (see `par`).
lty	line type; default is `1`, a solid line (see `par`).
...	optional arguments to be passed to the `lines` plotting function.

Details

In contrast to `abline`, this function plots only over the range of the observed x-values. The x-values are extracted from `mod` as the second column of the model matrix.

Value

`NULL`. This function is used for its side effect: adding a line to the plot.

Author(s)

John Fox <jfox@mcmaster.ca>

See Also

abline, lines

Examples

```
plot(repwt ~ weight, pch=c(1,2)[sex], data=Davis)
regLine(lm(repwt~weight, subset=sex=="M", data=Davis))
regLine(lm(repwt~weight, subset=sex=="F", data=Davis), lty=2)
```

---

residualPlots *Residual Plots and Curvature Tests for Linear Model Fits*

Description

Plots the residuals versus each term in a mean function and versus fitted values. Also computes a curvature test for each of the plots by adding a quadratic term and testing the quadratic to be zero. This is Tukey's test for nonadditivity when plotting against fitted values.

Usage

```
### This is a generic function with only one required argument:

residualPlots (model, ...)

## Default S3 method:
residualPlots(model, terms = ~., layout = NULL, ask,
                 main = "", fitted = TRUE, AsIs=TRUE, plot = TRUE,
                 tests = TRUE, ...)

## S3 method for class 'lm'
residualPlots(model, ...)

## S3 method for class 'glm'
residualPlots(model, ...)

### residualPlots calls residualPlot, so these arguments can be
### used with either function

residualPlot(model, ...)

## Default S3 method:
residualPlot(model, variable = "fitted", type = "pearson",
                 plot = TRUE,
                 quadratic = TRUE,
                 smoother=NULL, smoother.args=list(),
                 col.smooth=palette()[3],
                 labels,
```

```
                 id.method = "y",
                 id.n = if(id.method[1]=="identify") Inf else 0,
                 id.cex=1, id.col=palette()[1],
                 col = palette()[1], col.quad = palette()[2],
                 xlab, ylab, lwd = 1, lty = 1,
                 grid=TRUE, ...)

## S3 method for class 'lm'
residualPlot(model, ...)

## S3 method for class 'glm'
residualPlot(model, variable = "fitted", type = "pearson",
                 plot = TRUE, quadratic = FALSE,
                 smoother = loessLine, smoother.args=list(k=3), ...)
```

Arguments

- `model` A regression object.
- `terms` A one-sided formula that specifies a subset of the predictors. One residual plot is drawn for each specified. The default `~ .` is to plot against all predictors. For example, the specification `terms = ~ . - X3` would plot against all predictors except for `X3`. To get a plot against fitted values only, use the arguments `terms = ~ 1, fitted=TRUE`, Interactions are skipped. For polynomial terms, the plot is against the first-order variable (which may be centered and scaled depending on how the `poly` function is used). Plots against factors are boxplots. Plots against other matrix terms, like splines, use the result of `predict(model), type="terms")[, variable])` as the horizontal axis; if the `predict` method doesn't permit this type, then matrix terms are skipped.
- `layout` If set to a value like `c(1, 1)` or `c(4, 3)`, the layout of the graph will have this many rows and columns. If not set, the program will select an appropriate layout. If the number of graphs exceed nine, you must select the layout yourself, or you will get a maximum of nine per page. If `layout=NA`, the function does not set the layout and the user can use the `par` function to control the layout, for example to have plots from two models in the same graphics window.
- `ask` If `TRUE`, ask the user before drawing the next plot; if `FALSE`, don't ask.
- `main` Main title for the graphs. The default is `main=""` for no title.
- `fitted` If `TRUE`, the default, include the plot against fitted values.
- `AsIs` If `FALSE`, terms that use the "as-is" function `I` are skipped; if `TRUE`, the default, they are included.
- `plot` If `TRUE`, draw the plot(s).
- `tests` If `TRUE`, display the curvature tests. With glm's, the argument `start` is ignored in computing the curvature tests.
- `...` Additional arguments passed to `residualPlot` and then to `plot`.
- `variable` Quoted variable name for the horizontal axis, or `"fitted"` to plot versus fitted values.

type
: Type of residuals to be used. Pearson residuals are appropriate for `lm` objects since these are equivalent to ordinary residuals with ols and correctly weighted residuals with wls. Any quoted string that is an appropriate value of the `type` argument to `residuals.lm` or `"rstudent"` or `"rstandard"` for Studentized or standardized residuals.

quadratic
: if `TRUE`, fits the quadratic regression of the vertical axis on the horizontal axis and displays a lack of fit test. Default is `TRUE` for `lm` and FALSE for `glm`.

smoother
: the name of the smoother to use, selected from the choices descripbed at `ScatterplotSmoothers` For `lm` objects the default is `NULL`. For `glm` object the default is `loessLine`.

smoother.args
: arguments passed to the smoother. See `ScatterplotSmoothers`. For generalized linear models the number of elements in the spline basis is set to `k=3`; this is done to allow fitting for predictors with just a few support points. If you have many support points you may wish to set `k` to a higher number, or `k=-1` for the default used by `gam`.

col.smooth
: color for the smoother

id.method,labels,id.n,id.cex,id.col
: Arguments for the labelling of points. The default is `id.n=0` for labeling no points. See `showLabels` for details of these arguments.

col
: default color for points

col.quad
: default color for quadratic fit

xlab
: X-axis label. If not specified, a useful label is constructed by the function.

ylab
: Y-axis label. If not specified, a useful label is constructed by the function.

lwd
: line width for lines.

lty
: line type for quadratic.

grid
: If TRUE, the default, a light-gray background grid is put on the graph

Details

`residualPlots` draws one or more residuals plots depending on the value of the `terms` and `fitted` arguments. If `terms = ~ .`, the default, then a plot is produced of residuals versus each first-order term in the formula used to create the model. Interaction terms, spline terms, and polynomial terms of more than one predictor are skipped. In addition terms that use the "as-is" function, e.g., `I(X^2)`, will also be skipped unless you set the argument `AsIs=TRUE`. A plot of residuals versus fitted values is also included unless `fitted=FALSE`.

In addition to plots, a table of curvature tests is displayed. For plots against a term in the model formula, say `X1`, the test displayed is the t-test for for `I(X^2)` in the fit of `update, model, ~. + I(X^2))`. Econometricians call this a specification test. For factors, the displayed plot is a boxplot, and no curvature test is computed. For fitted values, the test is Tukey's one-degree-of-freedom test for nonadditivity. You can suppress the tests with the argument `tests=FALSE`.

`residualPlot`, which is called by `residualPlots`, should be viewed as an internal function, and is included here to display its arguments, which can be used with `residualPlots` as well. The `residualPlot` function returns the curvature test as an invisible result.

`residCurvTest` computes the curvature test only. For any factors a boxplot will be drawn. For any polynomials, plots are against the linear term. Other non-standard predictors like B-splines are skipped.

Value

For `lm` objects, returns a data.frame with one row for each plot drawn, one column for the curvature test statistic, and a second column for the corresponding p-value. This function is used primarily for its side effect of drawing residual plots.

Author(s)

Sanford Weisberg, <sandy@umn.edu>

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition. Sage.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition, Wiley, Chapter 8

See Also

See Also `lm`, `identify`, `showLabels`

Examples

```
residualPlots(lm(longley))
```

Robey *Fertility and Contraception*

Description

The `Robey` data frame has 50 rows and 3 columns. The observations are developing nations around 1990.

Usage

```
Robey
```

Format

This data frame contains the following columns:

region A factor with levels: `Africa`; `Asia`, Asia and Pacific; `Latin.Amer`, Latin America and Caribbean; `Near.East`, Near East and North Africa.

tfr Total fertility rate (children per woman).

contraceptors Percent of contraceptors among married women of childbearing age.

Source

Robey, B., Shea, M. A., Rutstein, O. and Morris, L. (1992) The reproductive revolution: New survey findings. *Population Reports*. Technical Report M-11.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

`Sahlins` *Agricultural Production in Mazulu Village*

Description

The `Sahlins` data frame has 20 rows and 2 columns. The observations are households in a Central African village.

Usage

```
Sahlins
```

Format

This data frame contains the following columns:

consumers Consumers/Gardener, ratio of consumers to productive individuals.

acres Acres/Gardener, amount of land cultivated per gardener.

Source

Sahlins, M. (1972) *Stone Age Economics.* Aldine [Table 3.1].

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

`Salaries` *Salaries for Professors*

Description

The 2008-09 nine-month academic salary for Assistant Professors, Associate Professors and Professors in a college in the U.S. The data were collected as part of the on-going effort of the college's administration to monitor salary differences between male and female faculty members.

Usage

```
Salaries
```

Format

A data frame with 397 observations on the following 6 variables.

`rank` a factor with levels `AssocProf` `AsstProf` `Prof`

`discipline` a factor with levels `A` ("theoretical" departments) or `B` ("applied" departments).

`yrs.since.phd` years since PhD.

`yrs.service` years of service.

`sex` a factor with levels `Female` `Male`

`salary` nine-month salary, in dollars.

References

Fox J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition Sage.

---

scatter3d — *Three-Dimensional Scatterplots and Point Identification*

Description

The `scatter3d` function uses the `rgl` package to draw 3D scatterplots with various regression surfaces. The function `identify3d` allows you to label points interactively with the mouse: Press the right mouse button (on a two-button mouse) or the centre button (on a three-button mouse), drag a rectangle around the points to be identified, and release the button. Repeat this procedure for each point or set of "nearby" points to be identified. To exit from point-identification mode, click the right (or centre) button in an empty region of the plot.

Usage

```
scatter3d(x, ...)

## S3 method for class 'formula'
scatter3d(formula, data, subset, radius, xlab, ylab, zlab, labels, ...)

## Default S3 method:
scatter3d(x, y, z,
  xlab=deparse(substitute(x)), ylab=deparse(substitute(y)),
  zlab=deparse(substitute(z)), axis.scales=TRUE, revolutions=0,
      bg.col=c("white", "black"),
  axis.col=if (bg.col == "white") c("darkmagenta", "black", "darkcyan")
  else c("darkmagenta", "white", "darkcyan"),
  surface.col=c("blue", "green", "orange", "magenta", "cyan", "red",
      "yellow", "gray"), surface.alpha=0.5,
  neg.res.col="red", pos.res.col="green",
  square.col=if (bg.col == "white") "black" else "gray", point.col="yellow",
  text.col=axis.col, grid.col=if (bg.col == "white") "black" else "gray",
```

```
  fogtype=c("exp2", "linear", "exp", "none"),
  residuals=(length(fit) == 1), surface=TRUE, fill=TRUE, grid=TRUE,
      grid.lines=26, df.smooth=NULL, df.additive=NULL,
  sphere.size=1, radius=1, threshold=0.01, speed=1, fov=60,
  fit="linear", groups=NULL, parallel=TRUE,
      ellipsoid=FALSE, level=0.5, ellipsoid.alpha=0.1,
      id.method=c("mahal", "xz", "y", "xyz", "identify", "none"),
  id.n=if (id.method == "identify") Inf else 0,
  labels=as.character(seq(along=x)), offset = ((100/length(x))^(1/3)) * 0.02,
  model.summary=FALSE, ...)

identify3d(x, y, z, axis.scales=TRUE, groups = NULL, labels = 1:length(x),
col = c("blue", "green", "orange", "magenta", "cyan", "red", "yellow", "gray"),
offset = ((100/length(x))^(1/3)) * 0.02)
```

Arguments

`formula`	"model" formula, of the form `y ~ x + z` or (to plot by groups) `y ~ x + z	g`, where `g` evaluates to a factor or other variable dividing the data into groups.
`data`	data frame within which to evaluate the formula.
`subset`	expression defining a subset of observations.
`x`	variable for horizontal axis.
`y`	variable for vertical axis (response).
`z`	variable for out-of-screen axis.
`xlab, ylab, zlab`	axis labels.
`axis.scales`	if `TRUE`, label the values of the ends of the axes. *Note:* For `identify3d` to work properly, the value of this argument must be the same as in `scatter3d`.
`revolutions`	number of full revolutions of the display.
`bg.col`	background colour; one of `"white"`, `"black"`.
`axis.col`	colours for axes; if `axis.scales` is `FALSE`, then the second colour is used for all three axes.
`surface.col`	vector of colours for regression planes, used in the order specified by `fit`; for multi-group plots, the colours are used for the regression surfaces and points in the several groups.
`surface.alpha`	transparency of regression surfaces, from `0.0` (fully transparent) to `1.0` (opaque); default is `0.5`.
`neg.res.col, pos.res.col`	colours for lines representing negative and positive residuals.
`square.col`	colour to use to plot squared residuals.
`point.col`	colour of points.
`text.col`	colour of axis labels.
`grid.col`	colour of grid lines on the regression surface(s).
`fogtype`	type of fog effect; one of `"exp2"`, `"linear"`, `"exp"`, `"none"`.

`residuals`	plot residuals if `TRUE`; if `residuals="squares"`, then the squared residuals are shown as squares (using code adapted from Richard Heiberger). Residuals are available only when there is one surface plotted.
`surface`	plot surface(s) (`TRUE` or `FALSE`).
`fill`	fill the plotted surface(s) with colour (`TRUE` or `FALSE`).
`grid`	plot grid lines on the regression surface(s) (`TRUE` or `FALSE`).
`grid.lines`	number of lines (default, 26) forming the grid, in each of the x and z directions.
`df.smooth`	degrees of freedom for the two-dimensional smooth regression surface; if `NULL` (the default), the `gam` function will select the degrees of freedom for a smoothing spline by generalized cross-validation; if a positive number, a fixed regression spline will be fit with the specified degrees of freedom.
`df.additive`	degrees of freedom for each explanatory variable in an additive regression; if `NULL` (the default), the `gam` function will select degrees of freedom for the smoothing splines by generalized cross-validation; if a positive number or a vector of two positive numbers, fixed regression splines will be fit with the specified degrees of freedom for each term.
`sphere.size`	general size of spheres representing points; the actual size is dependent on the number of observations.
`radius`	relative radii of the spheres representing the points. This is normally a vector of the same length as the variables giving the coordinates of the points, and for the `formula` method, that must be the case or the argument may be omitted, in which case spheres are the same size; for the `default` method, the default for the argument, `1`, produces spheres all of the same size. The radii are scaled so that their median is 1.
`threshold`	if the actual size of the spheres is less than the threshold, points are plotted instead.
`speed`	relative speed of revolution of the plot.
`fov`	field of view (in degrees); controls degree of perspective.
`fit`	one or more of `"linear"`, `"quadratic"`, `"smooth"`, `"additive"`; to display fitted surface(s); partial matching is supported – e.g., `c("lin", "quad")`.
`groups`	if `NULL` (the default), no groups are defined; if a factor, a different surface or set of surfaces is plotted for each level of the factor; in this event, the colours in `surface.col` are used successively for the points, surfaces, and residuals corresponding to each level of the factor.
`parallel`	when plotting surfaces by `groups`, should the surfaces be constrained to be parallel? A logical value, with default `TRUE`.
`ellipsoid`	plot concentration ellipsoid(s) (`TRUE` or `FALSE`).
`level`	expected proportion of bivariate-normal observations included in the concentration ellipsoid(s); default is 0.5.
`ellipsoid.alpha`	transparency of ellipsoids, from `0.0` (fully transparent) to `1.0` (opaque); default is `0.1`.

id.method	if "mahal" (the default), relatively extreme points are identified automatically according to their Mahalanobis distances from the centroid (point of means); if "identify", points are identified interactively by right-clicking and dragging a box around them; right-click in an empty area to exit from interactive-point-identification mode; if "xz", identify extreme points in the predictor plane; if "y", identify unusual values of the response; if "xyz" identify unusual values of an variable; if "none", no point identification. See showLabels for more information.
id.n	Number of relatively extreme points to identify automatically (default, 0 unless id.method="identify").
model.summary	print summary or summaries of the model(s) fit (TRUE or FALSE). scatter3d rescales the three variables internally to fit in the unit cube; this rescaling will affect regression coefficients.
labels	text labels for the points, one for each point; in the default method defaults to the observation indices, in the formula method to the row names of the data.
col	colours for the point labels, given by group. There must be at least as many colours as groups; if there are no groups, the first colour is used. Normally, the colours would correspond to the surface.col argument to scatter3d.
offset	vertical displacement for point labels (to avoid overplotting the points).
...	arguments to be passed down.

Value

scatter3d does not return a useful value; it is used for its side-effect of creating a 3D scatterplot. identify3d returns the labels of the identified points.

Note

You have to install the rgl package to produce 3D plots.

Author(s)

John Fox <jfox@mcmaster.ca>

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

See Also

rgl-package, gam

Examples

```
    if(interactive() && require(rgl) && require(mgcv)){
scatter3d(prestige ~ income + education, data=Duncan)
Sys.sleep(5) # wait 5 seconds
scatter3d(prestige ~ income + education | type, data=Duncan)
```

```
Sys.sleep(5)
scatter3d(prestige ~ income + education | type, surface=FALSE,
ellipsoid=TRUE, revolutions=3, data=Duncan)
scatter3d(prestige ~ income + education, fit=c("linear", "additive"),
data=Prestige)
Sys.sleep(5)
scatter3d(prestige ~ income + education | type,
    radius=(1 + women)^(1/3), data=Prestige)
}
## Not run:
# drag right mouse button to identify points, click right button in open area to exit
scatter3d(prestige ~ income + education, data=Duncan, id.method="identify")
scatter3d(prestige ~ income + education | type, data=Duncan, id.method="identify")

## End(Not run)
```

`scatterplot` *Scatterplots with Boxplots*

Description

Makes enhanced scatterplots, with boxplots in the margins, a nonparametric regression smooth, smoothed conditional spread, outlier identification, and a regression line; sp is an abbreviation for scatterplot.

Usage

```
scatterplot(x, ...)

## S3 method for class 'formula'
scatterplot(formula, data, subset, xlab, ylab, legend.title, legend.coords,
labels, ...)

## Default S3 method:
scatterplot(x, y,
    smoother=loessLine, smoother.args=list(), smooth, span,
    spread=!by.groups, reg.line=lm,
    boxplots=if (by.groups) "" else "xy",
    xlab=deparse(substitute(x)), ylab=deparse(substitute(y)), las=par("las"),
    lwd=1, lty=1,
    labels, id.method = "mahal",
    id.n = if(id.method[1]=="identify") length(x) else 0,
    id.cex = 1, id.col = palette()[1],
    log="", jitter=list(), xlim=NULL, ylim=NULL,
    cex=par("cex"), cex.axis=par("cex.axis"), cex.lab=par("cex.lab"),
    cex.main=par("cex.main"), cex.sub=par("cex.sub"),
    groups, by.groups=!missing(groups),
    legend.title=deparse(substitute(groups)), legend.coords,
```

```
    ellipse=FALSE, levels=c(.5, .95), robust=TRUE,
    col=if (n.groups == 1) palette()[3:1] else rep(palette(), length=n.groups),
    pch=1:n.groups,
    legend.plot=!missing(groups), reset.par=TRUE, grid=TRUE, ...)

sp(...)
```

Arguments

`x`	vector of horizontal coordinates
`y`	vector of vertical coordinates.
`formula`	a "model" formula, of the form `y ~ x` or (to plot by groups) `y ~ x | z`, where `z` evaluates to a factor or other variable dividing the data into groups. If `x` is a factor, then parallel boxplots are produced using the `Boxplot` function.
`data`	data frame within which to evaluate the formula.
`subset`	expression defining a subset of observations.
`smoother`	a function to draw a nonparametric-regression smooth; the default is `loessLine`, which does loess smoothing. The function `gamLine` fits a generalized additive model and allows including a link and error function. See `ScatterplotSmoothers`. Setting this argument to something other than a function, e.g., `FALSE` suppresses the smoother.
`smoother.args`	a list of named values to be passed to the smoother function; the specified elements of the list depend upon the smoother (see `ScatterplotSmoothers`).
`smooth, span`	these arguments are included for backwards compatility: if `smooth=TRUE` then `smoother` is set to `loessLine`, and if `span` is specified, it is added to `smoother.args`.
`spread`	if TRUE, estimate the (square root) of the variance function. For `loessLine` and for `gamLine`, this is done by separately smoothing the squares of the postive and negative residuals from the mean fit, and then adding the square root of the fitted values to the mean fit. For `quantregLine`, fit the .25 and .75 quantiles with a quantile regression additive model. The default is TRUE if `by.groups=FALSE` and FALSE is `by.groups=TRUE`.
`reg.line`	function to draw a regression line on the plot or `FALSE` not to plot a regression line.
`boxplots`	if `"x"` a boxplot for `x` is drawn below the plot; if `"y"` a boxplot for `y` is drawn to the left of the plot; if `"xy"` both boxplots are drawn; set to `""` or `FALSE` to suppress both boxplots.
`xlab`	label for horizontal axis.
`ylab`	label for vertical axis.
`las`	if `0`, ticks labels are drawn parallel to the axis; set to `1` for horizontal labels (see `par`).
`lwd`	width of linear-regression lines (default `1`).
`lty`	type of linear-regression lines (default `1`, solid line).

id.method,id.n,id.cex,id.col
: Arguments for the labelling of points. The default is id.n=0 for labeling no points. See showLabels for details of these arguments. If the plot uses different colors for groups, then the id.col argument is ignored and label colors are determined by the col argument.

labels
: a vector of point labels; if absent, the function tries to determine reasonable labels, and, failing that, will use observation numbers.

log
: same as the log argument to plot, to produce log axes.

jitter
: a list with elements x or y or both, specifying jitter factors for the horizontal and vertical coordinates of the points in the scatterplot. The jitter function is used to randomly perturb the points; specifying a factor of 1 produces the default jitter. Fitted lines are unaffected by the jitter.

xlim
: the x limits (min, max) of the plot; if NULL, determined from the data.

ylim
: the y limits (min, max) of the plot; if NULL, determined from the data.

groups
: a factor or other variable dividing the data into groups; groups are plotted with different colors and plotting characters.

by.groups
: if TRUE, regression lines are fit by groups.

legend.title
: title for legend box; defaults to the name of the groups variable.

legend.coords
: coordinates for placing legend; an be a list with components x and y to specify the coordinates of the upper-left-hand corner of the legend; or a quoted keyword, such as "topleft", recognized by legend.

ellipse
: if TRUE data-concentration ellipses are plotted.

levels
: level or levels at which concentration ellipses are plotted; the default is c(.5, .95).

robust
: if TRUE (the default) use the cov.trob function in the MASS package to calculate the center and covariance matrix for the data ellipses.

col
: colors for lines and points; the default is taken from the color palette, with palette()[3] for linear regression lines, palette()[2] for nonparametric regression lines, and palette()[1] for points if there are no groups, and successive colors for the groups if there are groups.

pch
: plotting characters for points; default is the plotting characters in order (see par).

cex, cex.axis, cex.lab, cex.main, cex.sub
: set sizes of various graphical elements; (see par).

legend.plot
: if TRUE then a legend for the groups is plotted in the upper margin.

reset.par
: if TRUE then plotting parameters are reset to their previous values when scatterplot exits; if FALSE then the mar and mfcol parameters are altered for the current plotting device. Set to FALSE if you want to add graphical elements (such as lines) to the plot.

...
: other arguments passed down and to plot.

grid
: If TRUE, the default, a light-gray background grid is put on the graph

Value

If points are identified, their labels are returned; otherwise NULL is returned invisibly.

Author(s)

John Fox <jfox@mcmaster.ca>

See Also

boxplot, jitter, legend, scatterplotMatrix, dataEllipse, Boxplot, cov.trob, showLabels, ScatterplotSmoothers.

Examples

```
scatterplot(prestige ~ income, data=Prestige, ellipse=TRUE)

if (interactive()){
scatterplot(prestige ~ income, data=Prestige, smoother=quantregLine)
}

scatterplot(prestige ~ income|type, data=Prestige, smoother=loessLine,
    smoother.args=list(span=1))

scatterplot(prestige ~ income|type, data=Prestige, legend.coords="topleft")

scatterplot(vocabulary ~ education, jitter=list(x=1, y=1),
data=Vocab, id.n=0, smoother=FALSE)

scatterplot(infant.mortality ~ gdp, log="xy", data=UN, id.n=5)

scatterplot(income ~ type, data=Prestige)

## Not run:
scatterplot(infant.mortality ~ gdp, id.method="identify", data=UN)

scatterplot(infant.mortality ~ gdp, id.method="identify", smoother=loessLine, data=UN)

## End(Not run)
```

---

scatterplotMatrix *Scatterplot Matrices*

Description

Enhanced scatterplot matrices with univariate displays down the diagonal; spm is an abbreviation for scatterplotMatrix. This function just sets up a call to pairs with custom panel functions.

Usage

```
scatterplotMatrix(x, ...)

## S3 method for class 'formula'
```

```
scatterplotMatrix(formula, data=NULL, subset, labels, ...)

## Default S3 method:
scatterplotMatrix(x, var.labels=colnames(x),
    diagonal=c("density", "boxplot", "histogram", "oned", "qqplot", "none"),
    adjust=1, nclass,
    plot.points=TRUE, smoother=loessLine, smoother.args=list(), smooth, span,
    spread = !by.groups, reg.line=lm,
    transform=FALSE, family=c("bcPower", "yjPower"),
    ellipse=FALSE, levels=c(.5, .95), robust=TRUE,
    groups=NULL, by.groups=FALSE,
    use=c("complete.obs", "pairwise.complete.obs"),
    labels, id.method="mahal", id.n=0, id.cex=1, id.col=palette()[1],
    col=if (n.groups == 1) palette()[3:1] else rep(palette(), length=n.groups),
    pch=1:n.groups, lwd=1, lty=1,
    cex=par("cex"), cex.axis=par("cex.axis"), cex.labels=NULL,
    cex.main=par("cex.main"),
    legend.plot=length(levels(groups)) > 1, row1attop=TRUE, ...)

spm(x, ...)
```

Arguments

`x` a data matrix, numeric data frame.

`formula` a one-sided "model" formula, of the form `~ x1 + x2 + ... + xk` or `~ x1 + x2 + ... + xk | z` where `z` evaluates to a factor or other variable to divide the data into groups.

`data` for `scatterplotMatrix.formula`, a data frame within which to evaluate the formula.

`subset` expression defining a subset of observations.

`labels,id.method,id.n,id.cex,id.col`
Arguments for the labelling of points. The default is `id.n=0` for labeling no points. See `showLabels` for details of these arguments. If the plot uses different colors for groups, then the `id.col` argument is ignored and label colors are determined by the `col` argument.

`var.labels` variable labels (for the diagonal of the plot).

`diagonal` contents of the diagonal panels of the plot.

`adjust` relative bandwidth for density estimate, passed to `density` function.

`nclass` number of bins for histogram, passed to `hist` function.

`plot.points` if `TRUE` the points are plotted in each off-diagonal panel.

`smoother` a function to draw a nonparametric-regression smooth; the default is `gamLine`, which uses the `gam` function in the **mgcv** package. For this and other smoothers, see `ScatterplotSmoothers`. Setting this argument to something other than a function, e.g., `FALSE` suppresses the smoother.

`smoother.args` a list of named values to be passed to the smoother function; the specified elements of the list depend upon the smoother (see `ScatterplotSmoothers`).

`smooth, span` these arguments are included for backwards compatility: if `smooth=TRUE` then `smoother` is set to `loessLine`, and if `span` is specified, it is added to `smoother.args`.

`spread` if TRUE, estimate the (square root) of the variance function. For `loessLine` and for `gamLine`, this is done by separately smoothing the squares of the postive and negative residuals from the mean fit, and then adding the square root of the fitted values to the mean fit. For `quantregLine`, fit the .25 and .75 quantiles with a quantile regression additive model. The default is TRUE if `by.groups=FALSE` and FALSE is `by.groups=TRUE`.

`reg.line` if not `FALSE` a line is plotted using the function given by this argument; e.g., using `rlm` in package `MASS` plots a robust-regression line.

`transform` if `TRUE`, multivariate normalizing power transformations are computed with `powerTransform`, rounding the estimated powers to 'nice' values for plotting; if a vector of powers, one for each variable, these are applied prior to plotting. If there are `groups` and `by.groups` is `TRUE`, then the transformations are estimated *conditional* on the `groups` factor.

`family` family of transformations to estimate: `"bcPower"` for the Box-Cox family or `"yjPower"` for the Yeo-Johnson family (see `powerTransform`).

`ellipse` if `TRUE` data-concentration ellipses are plotted in the off-diagonal panels.

`levels` levels or levels at which concentration ellipses are plotted; the default is `c(.5, .9)`.

`robust` if `TRUE` use the `cov.trob` function in the `MASS` package to calculate the center and covariance matrix for the data ellipses.

`groups` a factor or other variable dividing the data into groups; groups are plotted with different colors and plotting characters.

`by.groups` if `TRUE`, regression lines are fit by groups.

`use` if `"complete.obs"` (the default), cases with missing data are omitted; if `"pairwise.complete.obs"), a` `in each panel of the plot.`

`pch` plotting characters for points; default is the plotting characters in order (see `par`).

`col` colors for lines and points; the default is taken from the color palette, with `palette()[3]` for linear regression lines, `palette()[2]` for nonparametric regression lines, and `palette()[1]` for points if there are no groups, and successive colors for the groups if there are groups.

`lwd` width of linear-regression lines (default `1`).

`lty` type of linear-regression lines (default `1`, solid line).

`cex, cex.axis, cex.labels, cex.main`
set sizes of various graphical elements (see `par`).

`legend.plot` if `TRUE` then a legend for the groups is plotted in the first diagonal cell.

`row1attop` If `TRUE` (the default) the first row is at the top, as in a matrix, as opposed to at the bottom, as in graph (argument suggested by Richard Heiberger).

`...` arguments to pass down.

Value

`NULL`. This function is used for its side effect: producing a plot.

Author(s)

John Fox <jfox@mcmaster.ca>

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

See Also

`pairs`, `scatterplot`, `dataEllipse`, `powerTransform`, `bcPower`, `yjPower`, `cov.trob`, `showLabels`, `ScatterplotSmoothers`.

Examples

```
scatterplotMatrix(~ income + education + prestige | type, data=Duncan)
scatterplotMatrix(~ income + education + prestige,
    transform=TRUE, data=Duncan, smoother=loessLine)
scatterplotMatrix(~ income + education + prestige | type, smoother=FALSE,
by.group=TRUE, transform=TRUE, data=Duncan)
```

`ScatterplotSmoothers` *Smoothers to Draw Lines on Scatterplots*

Description

These smoothers are used to draw nonparametric-regression lines on scatterplots produced by the `scatterplot`, `scatterplotMatrix` and other `car` functions. The functions aren't meant to be called directly by the user, although the user can supply options via the `smoother.args` argument, the contents of which vary by the smoother (see *Details* below). The `gamLine` smoother uses the `gam` function in the **mgcv** package, the `loessLine` smoother uses the `loess` function in the **stats** package, and the `quantregLine` smoother uses the `rqss` function in the **quantreg** package.

Usage

```
gamLine(x, y, col, log.x, log.y, spread=FALSE, smoother.args, draw=TRUE)

loessLine(x, y, col, log.x, log.y, spread=FALSE, smoother.args, draw=TRUE)

quantregLine(x, y, col, log.x, log.y, spread=FALSE, smoother.args, draw=TRUE)
```

Arguments

`x`	x coordinates of points.
`y`	y coordinates of points.
`col`	line color.
`log.x`	`TRUE` if the x-axis is logged.

- `log.y` — TRUE if the y-axis is logged.
- `spread` — the default is to plot only an estimated mean or median. If this argument is TRUE, then a measure of spread is also plotted.
- `smoother.args` — additional options accapted by the smoother, in the form of a list of named values (see *Details* below).
- `draw` — if TRUE, the default, draw the smoother on the currently active graph. If FALSE, return a list with coordinates `x` and `y` for the points that make up the smooth and if requested `x.pos, y.pos, x.neg, y.neg` for the spread smooths.

Details

The function `loessLine` is a reimplementation of the `loess` smoother that has been used in `car` prior to September 2012. The only enhancement is the ability to set more arguments through the `smoother.args` argument.

The function `gamLine` is new and more general than the `loess` fitting because it allows fitting a generalized additive model using splines. You can specify a error distribution and link function.

The function `quantregLine` fits an additive model using splines with estimation based on L1 regression and quantile regression if you ask for the spread. It is likely to be more robust than the other smoothers.

The argument `smoother.args` is a list of named elements used to pass additional arguments to the smoother.

For `loessLine` the default value is `smoother.args=list(lty=1, lwd=2, lty.spread=2, lwd.spread=1, span=0.5,de` The arguments `lty` and `lwd` are the type and width respectively of the mean or median smooth, `smooth.lty` and `smooth.lwd` are the type and color of the spread smooths if requested. The arguments `span`, `degree` and `family` are passed to the `loess` function, `iterations=0` by default specifies no robustness iterations.

For `gamLine` the default is `smoother.args=list(lty=1, lwd=2, lty.spread=2, lwd.spread=1,k=-1, bs="tp", fam` The first for arguments are as for `loessLine`. The next two arguments are passed to the `gam` function to control the smoothing: `k=-1` allows `gam` to choose the number of splines in the basis function; `bs="tp"` provides the type of spline basis to be used with `"tp"` for the default thin-plate splines. The last three arguments allow providing a family, link and weights as in generalized linear models. See examples below.

For `quantregLine` the default is `smoother.args=list(lty=1, lwd=2, lty.spread=2, lwd.spread=1,lambda=IQR(x)` The first four arguments are as for `loessLine`. The last argument is passed to the `qss` function in `quantreg`. It is a smoothing parameter, here a robust estimate of the scale of the horizontal axis variable. This is an arbitrary choice, and may not work well in all circumstances.

Author(s)

John Fox <jfox@mcmaster.ca> and Sanford Weisberg<sandy@umn.edu>.

See Also

`scatterplot`, `scatterplotMatrix`, `gam`, `loess`, and `rqss`.

Examples

```
scatterplot(prestige ~ income, data=Prestige)
scatterplot(prestige ~ income, data=Prestige, smoother=gamLine)
scatterplot(prestige ~ income, data=Prestige, smoother=quantregLine)

scatterplot(prestige ~ income | type, data=Prestige)
scatterplot(prestige ~ income | type, data=Prestige, smoother=gamLine)
scatterplot(prestige ~ income | type, data=Prestige, smoother=quantregLine)
scatterplot(prestige ~ income | type, data=Prestige, smoother=NULL)

scatterplot(prestige ~ income | type, data=Prestige, spread=TRUE)
scatterplot(prestige ~ income | type, data=Prestige, smoother=gamLine, spread=TRUE)
scatterplot(prestige ~ income | type, data=Prestige, smoother=quantregLine, spread=TRUE)

scatterplot(weight ~ repwt | sex, spread=TRUE, data=Davis, smoother=loessLine)
scatterplot(weight ~ repwt | sex, spread=TRUE, data=Davis, smoother=gamLine) # messes up
scatterplot(weight ~ repwt | sex, spread=TRUE, data=Davis, smoother=quantregLine) #  robust
set.seed(12345)
w <- 1 + rpois(100, 5)
x <- rnorm(100)
p <- 1/(1 + exp(-(x + 0.5*x^2)))
s <- rbinom(100, w, p)
scatterplot(s/w ~ x, smoother=gamLine,
       smoother.args=list(family="binomial", weights=w))
scatterplot(s/w ~ x, smoother=gamLine,
       smoother.args=list(family=binomial, link="probit", weights=w))
scatterplot(s/w ~ x, smoother=gamLine,
       smoother.args=list(family=binomial, link="probit", weights=w))
scatterplot(s/w ~ x, smoother=loessLine, reg=FALSE)

y <- rbinom(100, 1, p)
scatterplot(y ~ x, smoother=gamLine, smoother.args=list(family=binomial))
```

---

showLabels   *Utility Functions to Identify and Mark Extreme Points in a 2D Plot.*

Description

This function is called by several graphical functions in the car package to mark extreme points in a 2D plot. Although the user is unlikely to call this function directly, the documentation below applies to all these other functions.

Usage

```
showLabels(x, y, labels=NULL, id.method="identify",
  id.n = length(x), id.cex=1, id.col=palette()[1], ...)
```

Arguments

`x`	Plotted horizontal coordinates.
`y`	Plotted vertical coordinates.
`labels`	Plotting labels. If `NULL`, case numbers will be used. If labels are long, the `substr` or `abbreviate` function can be used to shorten them.
`id.method`	How points are to be identified. See Details below.
`id.n`	Number of points to be identified. If set to zero, no points are identified.
`id.cex`	Controls the size of the plotted labels. The default is `1`.
`id.col`	Controls the color of the plotted labels.
`...`	additional arguments passed to `identify` or to `text`.

Details

The argument `id.method` determine how the points to be identified are selected. For the default value of `id.method="identify"`, the `identify` function is used to identify points interactively using the mouse. Up to `id.n` points can be identified, so if `id.n=0`, which is the default in many functions in the `car` package, then no point identification is done.

Automatic point identification can be done depending on the value of the argument `id.method`.

- `id.method = "x"` select points according to their value of `abs(x - mean(x))`
- `id.method = "y"` select points according to their value of `abs(y - mean(y))`
- `id.method = "mahal"` Treat `(x, y)` as if it were a bivariate sample, and select cases according to their Mahalanobis distance from `(mean(x), mean(y))`
- `id.method` can be a vector of the same length as `x` consisting of values to determine the points to be labeled. For example, for a linear model `m`, setting `id.method=cooks.distance(m), id.n=4` will label the points corresponding to the four largest values of Cook's distance, or `id.method = abs(residuals(m, t` would label the two observations corresponding to the largest absolute Pearson residuals. Warning: If missing data are present, points may be incorrectly labelled.
- `id.method` can be a vector of case numbers or case-labels, in which case those cases will be labeled. Warning: If missing data are present, a list of case numbers may identify the wrong points. A list of case labels, however, will work correctly with missing values.

With `showLabels`, the `id.method` argument can be a list, so, for example `id.method=list("x", "y")` would label according to the horizontal and vertical axes variables.

Finally, if the axes in the graph are logged, the function uses logged-variables where appropriate.

Value

A utility function primarily used for its side-effect of drawing labels on a plot. Returns invisibly the labels of the selected points, or NULL if no points are selected. Although intended for use with other functions in the car package, this function can be used directly.

Author(s)

John Fox <jfox@mcmaster.ca>, Sanford Weisberg <sandy@umn.edu>

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition, Wiley.

See Also

avPlots, residualPlots, crPlots, leveragePlots

Examples

```
plot(income ~ education, Prestige)
with(Prestige, showLabels(education, income,
     labels = rownames(Prestige), id.method=list("x", "y"), id.n=3))
m <- lm(income ~ education, Prestige)
plot(income ~ education, Prestige)
abline(m)
with(Prestige, showLabels(education, income,
     labels=rownames(Prestige), id.method=abs(residuals(m)), id.n=4))
```

---

sigmaHat *Return the scale estimate for a regression model*

Description

This function provides a consistent method to return the estimated scale from a linear, generalized linear, nonlinear, or other model.

Usage

```
sigmaHat(object)
```

Arguments

object	A regression object of type lm, glm or nls

Details

For an lm or nls object, the returned quantity is the square root of the estimate of σ. For a glm object, the returned quantity is the square root of the estimated dispersion parameter.

Value

A nonnegative number

Author(s)

Sanford Weisberg, <sandy@umn.edu>

Examples

```
m1 <- lm(prestige ~ income + education, data=Duncan)
sigmaHat(m1)
```

SLID *Survey of Labour and Income Dynamics*

Description

The SLID data frame has 7425 rows and 5 columns. The data are from the 1994 wave of the Canadian Survey of Labour and Income Dynamics, for the province of Ontario. There are missing data, particularly for wages.

Usage

```
SLID
```

Format

This data frame contains the following columns:

wages Composite hourly wage rate from all jobs.

education Number of years of schooling.

age in years.

sex A factor with levels: Female, Male.

language A factor with levels: English, French, Other.

Source

The data are taken from the public-use dataset made available by Statistics Canada, and prepared by the Institute for Social Research, York University.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Soils *Soil Compositions of Physical and Chemical Characteristics*

Description

Soil characteristics were measured on samples from three types of contours (Top, Slope, and Depression) and at four depths (0-10cm, 10-30cm, 30-60cm, and 60-90cm). The area was divided into 4 blocks, in a randomized block design. (Suggested by Michael Friendly.)

Usage

```
Soils
```

Format

A data frame with 48 observations on the following 14 variables. There are 3 factors and 9 response variables.

`Group` a factor with 12 levels, corresponding to the combinations of `Contour` and `Depth`

`Contour` a factor with 3 levels: `Depression` `Slope` `Top`

`Depth` a factor with 4 levels: `0-10` `10-30` `30-60` `60-90`

`Gp` a factor with 12 levels, giving abbreviations for the groups: `D0` `D1` `D3` `D6` `S0` `S1` `S3` `S6` `T0` `T1` `T3` `T6`

`Block` a factor with levels `1` `2` `3` `4`

`pH` soil pH

`N` total nitrogen in %

`Dens` bulk density in gm/cm^3

`P` total phosphorous in ppm

`Ca` calcium in me/100 gm.

`Mg` magnesium in me/100 gm.

`K` phosphorous in me/100 gm.

`Na` sodium in me/100 gm.

`Conduc` conductivity

Details

These data provide good examples of MANOVA and canonical discriminant analysis in a somewhat complex multivariate setting. They may be treated as a one-way design (ignoring `Block`), by using either `Group` or `Gp` as the factor, or a two-way randomized block design using `Block`, `Contour` and `Depth` (quantitative, so orthogonal polynomial contrasts are useful).

Source

Horton, I. F.,Russell, J. S., and Moore, A. W. (1968) Multivariate-covariance and canonical analysis: A method for selecting the most effective discriminators in a multivariate situation. *Biometrics* **24**, 845–858. http://www.stat.lsu.edu/faculty/moser/exst7037/soils.sas

References

Khattree, R., and Naik, D. N. (2000) *Multivariate Data Reduction and Discrimination with SAS Software.* SAS Institute.

Friendly, M. (2006) Data ellipses, HE plots and reduced-rank displays for multivariate linear models: SAS software and examples. *Journal of Statistical Software*, 17(6), http://www.jstatsoft.org/v17/i06.

`some` *Sample a Few Elements of an Object*

Description

Randomly select a few elements of an object, typically a data frame, matrix, vector, or list. If the object is a data frame or a matrix, then rows are sampled.

Usage

```
some(x, ...)

## S3 method for class 'data.frame'
some(x, n=10, ...)

## S3 method for class 'matrix'
some(x, n=10, ...)

## Default S3 method:
some(x, n=10, ...)
```

Arguments

`x`	the object to be sampled.
`n`	number of elements to sample.
`...`	arguments passed down.

Value

Sampled elements or rows.

Note

These functions are adapted from `head` and `tail` in the `utils` package.

Author(s)

John Fox <jfox@mcmaster.ca>

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

See Also

head, tail.

Examples

```
some(Duncan)
```

---

spreadLevelPlot *Spread-Level Plots*

Description

Creates plots for examining the possible dependence of spread on level, or an extension of these plots to the studentized residuals from linear models.

Usage

```
spreadLevelPlot(x, ...)

slp(...)

## S3 method for class 'formula'
spreadLevelPlot(x, data=NULL, subset, na.action,
    main=paste("Spread-Level Plot for", varnames[response],
    "by", varnames[-response]), ...)

## Default S3 method:
spreadLevelPlot(x, by, robust.line=TRUE,
start=0, xlab="Median", ylab="Hinge-Spread", point.labels=TRUE, las=par("las"),
main=paste("Spread-Level Plot for", deparse(substitute(x)),
"by", deparse(substitute(by))), col=palette()[1], col.lines=palette()[2],
    pch=1, lwd=2, grid=TRUE, ...)

## S3 method for class 'lm'
spreadLevelPlot(x, robust.line=TRUE,
xlab="Fitted Values",
ylab="Absolute Studentized Residuals", las=par("las"),
main=paste("Spread-Level Plot for\n", deparse(substitute(x))),
pch=1, col=palette()[1], col.lines=palette()[2], lwd=2, grid=TRUE, ...)
```

```
## S3 method for class 'spreadLevelPlot'
print(x, ...)
```

Arguments

x	a formula of the form y ~ x, where y is a numeric vector and x is a factor, or an lm object to be plotted; alternatively a numeric vector.
data	an optional data frame containing the variables to be plotted. By default the variables are taken from the environment from which spreadLevelPlot is called.
subset	an optional vector specifying a subset of observations to be used.
na.action	a function that indicates what should happen when the data contain NAs. The default is set by the na.action setting of options.
by	a factor, numeric vector, or character vector defining groups.
robust.line	if TRUE a robust line is fit using the rlm function in the MASS package; if FALSE a line is fit using lm.
start	add the constant start to each data value.
main	title for the plot.
xlab	label for horizontal axis.
ylab	label for vertical axis.
point.labels	if TRUE label the points in the plot with group names.
las	if 0, ticks labels are drawn parallel to the axis; set to 1 for horizontal labels (see par).
col	color for points; the default is the first entry in the current color palette (see palette and par).
col.lines	color for lines; default is the second entry in the current palette
pch	plotting character for points; default is 1 (a circle, see par).
lwd	line width; default is 2 (see par).
grid	If TRUE, the default, a light-gray background grid is put on the graph
...	arguments passed to plotting functions.

Details

Except for linear models, computes the statistics for, and plots, a Tukey spread-level plot of log(hinge-spread) vs. log(median) for the groups; fits a line to the plot; and calculates a spread-stabilizing transformation from the slope of the line.

For linear models, plots log(abs(studentized residuals) vs. log(fitted values).

The function slp is an abbreviation for spreadLevelPlot.

Value

An object of class `spreadLevelPlot` containing:

`Statistics` a matrix with the lower-hinge, median, upper-hinge, and hinge-spread for each group. (Not for an `lm` object.)

`PowerTransformation` spread-stabilizing power transformation, calculated as $1 - slope$ of the line fit to the plot.

Author(s)

John Fox <jfox@mcmaster.ca>

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Hoaglin, D. C., Mosteller, F. and Tukey, J. W. (Eds.) (1983) *Understanding Robust and Exploratory Data Analysis.* Wiley.

See Also

`hccm`, `ncvTest`

Examples

```
spreadLevelPlot(interlocks + 1 ~ nation, data=Ornstein)
slp(lm(interlocks + 1 ~ assets + sector + nation, data=Ornstein))
```

States — *Education and Related Statistics for the U.S. States*

Description

The `States` data frame has 51 rows and 8 columns. The observations are the U. S. states and Washington, D. C.

Usage

```
States
```

Format

This data frame contains the following columns:

region U. S. Census regions. A factor with levels: `ENC`, East North Central; `ESC`, East South Central; `MA`, Mid-Atlantic; `MTN`, Mountain; `NE`, New England; `PAC`, Pacific; `SA`, South Atlantic; `WNC`, West North Central; `WSC`, West South Central.

pop Population: in 1,000s.

SATV Average score of graduating high-school students in the state on the *verbal* component of the Scholastic Aptitude Test (a standard university admission exam).

SATM Average score of graduating high-school students in the state on the *math* component of the Scholastic Aptitude Test.

percent Percentage of graduating high-school students in the state who took the SAT exam.

dollars State spending on public education, in \$1000s per student.

pay Average teacher's salary in the state, in $1000s.

Source

United States (1992) *Statistical Abstract of the United States.* Bureau of the Census.

References

Moore, D. (1995) *The Basic Practice of Statistics*. Freeman, Table 2.1.

`subsets` *Plot Output from regsubsets Function in leaps package*

Description

The `regsubsets` function in the **leaps** package finds optimal subsets of predictors. This function plots a measure of fit (see the `statistic` argument below) against subset size.

Usage

```
subsets(object, ...)

## S3 method for class 'regsubsets'
subsets(object,
    names=abbreviate(object$xnames, minlength = abbrev),
    abbrev=1, min.size=1, max.size=length(names), legend,
    statistic=c("bic", "cp", "adjr2", "rsq", "rss"),
    las=par('las'), cex.subsets=1, ...)
```

Arguments

`object`	a `regsubsets` object produced by the `regsubsets` function in the **leaps** package.
`names`	a vector of (short) names for the predictors, excluding the regression intercept, if one is present; if missing, these are derived from the predictor names in `object`.
`abbrev`	minimum number of characters to use in abbreviating predictor names.
`min.size`	minimum size subset to plot; default is `1`.
`max.size`	maximum size subset to plot; default is number of predictors.
`legend`	`TRUE` to plot a legend of predictor names; defaults to `TRUE` if abbreviations are computed for predictor names. The legend is placed on the plot interactively with the mouse. By expanding the left or right plot margin, you can place the legend in the margin, if you wish (see `par`).
`statistic`	statistic to plot for each predictor subset; one of: `"bic"`, Bayes Information Criterion; `"cp"`, Mallows's C_p; `"adjr2"`, R^2 adjusted for degrees of freedom; `"rsq"`, unadjusted R^2; `"rss"`, residual sum of squares.
`las`	if `0`, ticks labels are drawn parallel to the axis; set to `1` for horizontal labels (see `par`).
`cex.subsets`	can be used to change the relative size of the characters used to plot the regression subsets; default is `1`.
`...`	arguments to be passed down to `subsets.regsubsets` and `plot`.

Value

`NULL` if the `legend` is `TRUE`; otherwise a data frame with the legend.

Author(s)

John Fox

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

See Also

`regsubsets`

Examples

```
if (interactive() && require(leaps)){
subsets(regsubsets(undercount ~ ., data=Ericksen))
}
```

symbox *Boxplots for transformations to symmetry*

Description

symbox first transforms x to each of a series of selected powers, with each transformation standardized to mean 0 and standard deviation 1. The results are then displayed side-by-side in boxplots, permiting a visual assessment of which power makes the distribution reasonably symmetric.

Usage

```
symbox(x, ...)
## S3 method for class 'formula'
symbox(formula, data=NULL, subset, na.action=NULL, ylab,  ...)
## Default S3 method:
symbox(x, powers = c(-1, -0.5, 0, 0.5, 1), start=0,
trans=bcPower, xlab="Powers", ylab, ...)
```

Arguments

x	a numeric vector.
formula	a one-sided formula specifying a single numeric variable.
data, subset, na.action	as for statistical modeling functions (see, e.g., lm).
xlab, ylab	axis labels; if ylab is missing, a label will be supplied.
powers	a vector of selected powers to which x is to be raised. For meaningful comparison of powers, 1 should be included in the vector of powers.
start	a constant to be added to x.
trans	a transformation function whose first argument is a numeric vector and whose second argument is a transformation parameter, given by the powers argument; the default is bcPower, and another possibility is yjPower.
...	arguments to be passed down.

Value

as returned by boxplot.

Author(s)

Gregor Gorjanc, John Fox <jfox@mcmaster.ca>, and Sanford Weisberg.

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition. Sage.

See Also

boxplot, boxcox, bcPower, yjPower

Examples

```
symbox(~ income, data=Prestige)
```

---

testTransform *Likelihood-Ratio Tests for Univariate or Multivariate Power Transformations to Normality*

Description

testTransform computes likelihood ratio tests for particular transformations based on powerTransform objects.

Usage

```
testTransform(object, lambda)

## S3 method for class 'powerTransform'
testTransform(object, lambda=rep(1, dim(object$y)[2]))
```

Arguments

object	An object created by a call to estimateTransform or powerTransform.
lambda	A vector of values of length equal to the number of variables to be transformed.

Details

The function powerTransform is used to estimate a power transformation for a univariate or multivariate sample or multiple linear regression problem, using the method of Box and Cox (1964). It is usual to round the estimates to nearby convenient values, and this function is use to compulte a likelihood ratio test for values of the transformation parameter other than the ml estimate. This is a generic function, but with only one method, for objects of class powerTransform.

Value

A data frame with one row giving the value of the test statistic, its degrees of freedom, and a p-value. The test is the likelihood ratio test, comparing the value of the log-likelihood at the hypothesized value to the value of the log-likelihood at the maximum likelihood estimate.

Author(s)

Sanford Weisberg, <sandy@umn.edu>

References

Box, G. E. P. and Cox, D. R. (1964) An analysis of transformations. *Journal of the Royal Statisistical Society, Series B*. 26 211-46.

Cook, R. D. and Weisberg, S. (1999) *Applied Regression Including Computing and Graphics*. Wiley.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition. Wiley.

See Also

powerTransform.

Examples

```
summary(a3 <- powerTransform(cbind(len, ADT, trks, sigs1) ~ hwy, Highway1))
# test lambda = (0 0 0 -1)
testTransform(a3, c(0, 0, 0, -1))
```

Transact *Transaction data*

Description

Data on transaction times in branch offices of a large Australian bank.

Usage

```
Transact
```

Format

This data frame contains the following columns:

t1 number of type 1 transactions

t2 number of type 2 transactions

time total transaction time, minutes

Source

Cunningham, R. and Heathcote, C. (1989), Estimating a non-Gaussian regression model with multicollinearity. Australian Journal of Statistics, 31,12-17.

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition. Wiley, Section 4.6.1.

TransformationAxes *Axes for Transformed Variables*

Description

These functions produce axes for the original scale of transformed variables. Typically these would appear as additional axes to the right or at the top of the plot, but if the plot is produced with `axes=FALSE`, then these functions could be used for axes below or to the left of the plot as well.

Usage

```
basicPowerAxis(power, base=exp(1),
    side=c("right", "above", "left", "below"),
    at, start=0, lead.digits=1, n.ticks, grid=FALSE, grid.col=gray(0.50),
    grid.lty=2,
    axis.title="Untransformed Data", cex=1, las=par("las"))

bcPowerAxis(power, side=c("right", "above", "left", "below"),
    at, start=0, lead.digits=1, n.ticks, grid=FALSE, grid.col=gray(0.50),
    grid.lty=2,
    axis.title="Untransformed Data", cex=1, las=par("las"))

yjPowerAxis(power, side=c("right", "above", "left", "below"),
at, lead.digits=1, n.ticks, grid=FALSE, grid.col=gray(0.50),
  grid.lty=2,
axis.title="Untransformed Data", cex=1, las=par("las"))

probabilityAxis(scale=c("logit", "probit"),
side=c("right", "above", "left", "below"),
at, lead.digits=1, grid=FALSE, grid.lty=2, grid.col=gray(0.50),
    axis.title = "Probability", interval = 0.1, cex = 1, las=par("las"))
```

Arguments

`power`	power for Box-Cox, Yeo-Johnson, or simple power transformation.
`scale`	transformation used for probabilities, `"logit"` (the default) or `"probit"`.
`side`	side at which the axis is to be drawn; numeric codes are also permitted: `side = 1` for the bottom of the plot, `side=2` for the left side, `side = 3` for the top, `side = 4` for the right side.
`at`	numeric vector giving location of tick marks on original scale; if missing, the function will try to pick nice locations for the ticks.
`start`	if a *start* was added to a variable (e.g., to make all data values positive), it can now be subtracted from the tick labels.
`lead.digits`	number of leading digits for determining 'nice' numbers for tick labels (default is `1`.

`n.ticks`	number of tick marks; if missing, same as corresponding transformed axis.
`grid`	if `TRUE` grid lines for the axis will be drawn.
`grid.col`	color of grid lines.
`grid.lty`	line type for grid lines.
`axis.title`	title for axis.
`cex`	relative character expansion for axis label.
`las`	if `0`, ticks labels are drawn parallel to the axis; set to `1` for horizontal labels (see `par`).
`base`	base of log transformation for `power.axis` when `power = 0`.
`interval`	desired interval between tick marks on the probability scale.

Details

The transformations corresponding to the three functions are as follows:

`basicPowerAxis`**:** Simple power transformation, $x' = x^p$ for $p \neq 0$ and $x' = \log x$ for $p = 0$.

`bcPowerAxis`**:** Box-Cox power transformation, $x' = (x^\lambda - 1)/\lambda$ for $\lambda \neq 0$ and $x' = \log x$ for $\lambda = 0$.

`yjPowerAxis`**:** Yeo-Johnson power transformation, for non-negative x, the Box-Cox transformation of $x + 1$; for negative x, the Box-Cox transformation of $|x| + 1$ with power $2 - p$.

`probabilityAxis`**:** logit or probit transformation, $\text{logit} = \log[p/(1-p)]$, or $\text{probit} = \Phi^{-1}(p)$, where Φ^{-1} is the standard-normal quantile function.

These functions will try to place tick marks at reasonable locations, but producing a good-looking graph sometimes requires some fiddling with the `at` argument.

Value

These functions are used for their side effects: to draw axes.

Author(s)

John Fox <`jfox@mcmaster.ca`>

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

See Also

`basicPower`, `bcPower`, `yjPower`, `logit`.

Examples

```
UN <- na.omit(UN)
par(mar=c(5, 4, 4, 4) + 0.1) # leave space on right

with(UN, plot(log(gdp, 10), log(infant.mortality, 10)))
basicPowerAxis(0, base=10, side="above",
  at=c(50, 200, 500, 2000, 5000, 20000), grid=TRUE,
  axis.title="GDP per capita")
basicPowerAxis(0, base=10, side="right",
  at=c(5, 10, 20, 50, 100), grid=TRUE,
  axis.title="infant mortality rate per 1000")

with(UN, plot(bcPower(gdp, 0), bcPower(infant.mortality, 0)))
bcPowerAxis(0, side="above",
  grid=TRUE, axis.title="GDP per capita")
bcPowerAxis(0, side="right",
  grid=TRUE, axis.title="infant mortality rate per 1000")

with(UN, qqPlot(logit(infant.mortality/1000)))
probabilityAxis()

with(UN, qqPlot(qnorm(infant.mortality/1000)))
probabilityAxis(at=c(.005, .01, .02, .04, .08, .16), scale="probit")
```

UN *GDP and Infant Mortality*

Description

The UN data frame has 207 rows and 2 columns. The data are for 1998 and are from the United Nations; the observations are nations of the world. There are some missing data.

Usage

```
UN
```

Format

This data frame contains the following columns:

infant.mortality Infant morality rate, infant deaths per 1000 live births.

gdp GDP per capita, in U.S.~dollars.

Source

United Nations (1998) Social indicators. http://www.un.org/Depts/unsd/social/main.htm.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

USPop — *Population of the United States*

Description

The `USPop` data frame has 22 rows and 1 columns. This is a decennial time-series, from 1790 to 2000.

Usage

```
USPop
```

Format

This data frame contains the following columns:

year census year.

population Population in millions.

Source

U.S.~Census Bureau: http://www.census-charts.com/Population/pop-us-1790-2000.html, downloaded 1 May 2008.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

---

vif — *Variance Inflation Factors*

Description

Calculates variance-inflation and generalized variance-inflation factors for linear and generalized linear models.

Usage

```
vif(mod, ...)

## S3 method for class 'lm'
vif(mod, ...)
```

Arguments

`mod`	an object that inherits from class `lm`, such as an `lm` or `glm` object.
`...`	not used.

Details

If all terms in an unweighted linear model have 1 df, then the usual variance-inflation factors are calculated.

If any terms in an unweighted linear model have more than 1 df, then generalized variance-inflation factors (Fox and Monette, 1992) are calculated. These are interpretable as the inflation in size of the confidence ellipse or ellipsoid for the coefficients of the term in comparison with what would be obtained for orthogonal data.

The generalized vifs are invariant with respect to the coding of the terms in the model (as long as the subspace of the columns of the model matrix pertaining to each term is invariant). To adjust for the dimension of the confidence ellipsoid, the function also prints $GVIF^{1/(2\times df)}$ where df is the degrees of freedom associated with the term.

Through a further generalization, the implementation here is applicable as well to other sorts of models, in particular weighted linear models and generalized linear models, that inherit from class `lm`.

Value

A vector of vifs, or a matrix containing one row for each term in the model, and columns for the GVIF, df, and $GVIF^{1/(2\times df)}$.

Author(s)

Henric Nilsson and John Fox `<jfox@mcmaster.ca>`

References

Fox, J. and Monette, G. (1992) Generalized collinearity diagnostics. *JASA*, **87**, 178–183.

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Examples

```
vif(lm(prestige ~ income + education, data=Duncan))
vif(lm(prestige ~ income + education + type, data=Duncan))
```

Vocab *Vocabulary and Education*

Description

The `Vocab` data frame has 21,638 rows and 5 columns. The observations are respondents to U.S. General Social Surveys, 1972-2004.

Usage

```
Vocab
```

Format

This data frame contains the following columns:

year Year of the survey.

sex Sex of the respondent, `Female` or `Male`.

education Education, in years.

vocabulary Vocabulary test score: number correct on a 10-word test.

Source

National Opinion Research Center *General Social Survey.* GSS Cumulative Datafile 1972-2004, downloaded from http://sda.berkeley.edu/archive.htm.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

wcrossprod *Weighted Matrix Crossproduct*

Description

Given matrices `x` and `y` as arguments and an optional matrix or vector of weights, `w`, return a weighted matrix cross-product, `t(x) w y`. If no weights are supplied, or the weights are constant, the function uses `crossprod` for speed.

Usage

```
wcrossprod(x, y, w)
```

Arguments

x,y	x, y numeric matrices; missing(y) is taken to be the same matrix as x. Vectors are promoted to single-column or single-row matrices, depending on the context.
w	A numeric vector or matrix of weights, conformable with x and y.

Value

A numeric matrix, with appropriate dimnames taken from x and y.

Author(s)

Michael Friendly, John Fox <jfox@mcmaster.ca>

See Also

crossprod

Examples

```
set.seed(12345)
n <- 24
drop <- 4
sex <- sample(c("M", "F"), n, replace=TRUE)
x1 <- 1:n
x2 <- sample(1:n)
extra <- c( rep(0, n - drop), floor(15 + 10 * rnorm(drop)) )
y1 <- x1 + 3*x2 + 6*(sex=="M") + floor(10 * rnorm(n)) + extra
y2 <- x1 - 2*x2 - 8*(sex=="M") + floor(10 * rnorm(n)) + extra
# assign non-zero weights to 'dropped' obs
wt <- c(rep(1, n-drop), rep(.2,drop))

X <- cbind(x1, x2)
Y <- cbind(y1, y2)
wcrossprod(X)
wcrossprod(X, w=wt)

wcrossprod(X, Y)
wcrossprod(X, Y, w=wt)

wcrossprod(x1, y1)
wcrossprod(x1, y1, w=wt)
```

WeightLoss *Weight Loss Data*

Description

Contrived data on weight loss and self esteem over three months, for three groups of individuals: Control, Diet and Diet + Exercise. The data constitute a double-multivariate design.

Usage

```
WeightLoss
```

Format

A data frame with 34 observations on the following 7 variables.

- `group` a factor with levels `Control` `Diet` `DietEx`.
- `wl1` Weight loss at 1 month
- `wl2` Weight loss at 2 months
- `wl3` Weight loss at 3 months
- `se1` Self esteem at 1 month
- `se2` Self esteem at 2 months
- `se3` Self esteem at 3 months

Details

Helmert contrasts are assigned to `group`, comparing `Control` vs. (`Diet` `DietEx`) and `Diet` vs. `DietEx`.

Source

Originally taken from `http://www.csun.edu/~ata20315/psy524/main.htm`, but modified slightly. Courtesy of Michael Friendly.

`which.names` *Position of Row Names*

Description

These functions return the indices of row names in a data frame or a vector of names. `whichNames` is just an alias for `which.names`.

Usage

```
which.names(names, object)
whichNames(...)
```

Arguments

`names`	a name or character vector of names.
`object`	a data frame or character vector of (row) names.
`...`	arguments to be passed to `which.names`.

Value

Returns the index or indices of `names` within `object`.

Author(s)

John Fox <jfox@mcmaster.ca>

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Examples

```
which.names(c('minister', 'conductor'), Duncan)
## [1]  6 16
```

Womenlf — *Canadian Women's Labour-Force Participation*

Description

The `Womenlf` data frame has 263 rows and 4 columns. The data are from a 1977 survey of the Canadian population.

Usage

```
Womenlf
```

Format

This data frame contains the following columns:

partic Labour-Force Participation. A factor with levels (note: out of order): `fulltime`, Working full-time; `not.work`, Not working outside the home; `parttime`, Working part-time.

hincome Husband's income, $1000s.

children Presence of children in the household. A factor with levels: `absent`, `present`.

region A factor with levels: `Atlantic`, Atlantic Canada; `BC`, British Columbia; `Ontario`; `Prairie`, Prairie provinces; `Quebec`.

Source

Social Change in Canada Project. York Institute for Social Research.

References

Fox, J. (2008) *Applied Regression Analysis and Generalized Linear Models*, Second Edition. Sage.

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

`Wool` *Wool data*

Description

This is a three-factor experiment with each factor at three levels, for a total of 27 runs. Samples of worsted yarn were with different levels of the three factors were given a cyclic load until the sample failed. The goal is to understand how cycles to failure depends on the factors.

Usage

```
Wool
```

Format

This data frame contains the following columns:

len length of specimen (250, 300, 350 mm)

amp amplitude of loading cycle (8, 9, 10 min)

load load (40, 45, 50g)

cycles number of cycles until failure

Source

Box, G. E. P. and Cox, D. R. (1964). An analysis of transformations (with discussion). *J. Royal Statist. Soc.*, B26, 211-46.

References

Fox, J. and Weisberg, S. (2011) *An R Companion to Applied Regression*, Second Edition, Sage.

Weisberg, S. (2005) *Applied Linear Regression*, Third Edition. Wiley, Section 6.3.

Index

*Topic **algebra**
wcrossprod, 147
*Topic **aplot**
Ellipses, 53
panel.car, 99
regLine, 110
TransformationAxes, 142
*Topic **array**
wcrossprod, 147
*Topic **datasets**
Adler, 5
AMSsurvey, 5
Angell, 6
Anscombe, 12
Baumann, 16
Bfox, 18
Blackmoor, 19
Burt, 28
CanPop, 28
Chile, 34
Chirot, 35
Cowles, 38
Davis, 41
DavisThin, 42
Depredations, 48
Duncan, 51
Ericksen, 56
Florida, 59
Freedman, 60
Friendly, 61
Ginzberg, 62
Greene, 62
Guyer, 63
Hartnagel, 64
Highway1, 66
Leinhardt, 76
Mandel, 87
Migration, 88
Moore, 92
Mroz, 93
OBrienKaiser, 95
Ornstein, 96
Pottery, 101
Prestige, 104
Quartet, 108
Robey, 114
Sahlins, 115
Salaries, 115
SLID, 131
Soils, 132
States, 136
Transact, 141
UN, 144
USPop, 145
Vocab, 147
WeightLoss, 148
Womenlf, 150
Wool, 151
*Topic **distribution**
qqPlot, 105
*Topic **hplot**
avPlots, 13
Boxplot, 25
ceresPlots, 31
crPlots, 39
densityPlot, 46
dfbetaPlots, 49
Ellipses, 53
infIndexPlot, 70
invResPlot, 72
invTranPlot, 74
leveragePlots, 78
mmps, 89
plot.powerTransform, 100
residualPlots, 111
scatter3d, 116
scatterplot, 120
scatterplotMatrix, 123

ScatterplotSmoothers, 126
spreadLevelPlot, 134
subsets, 137
symbox, 139
∗Topic **htest**
Anova, 7
leveneTest, 77
linearHypothesis, 80
ncvTest, 94
outlierTest, 97
∗Topic **interface**
carWeb, 30
∗Topic **manip**
boxCoxVariable, 24
logit, 86
recode, 108
∗Topic **models**
Anova, 7
Contrasts, 36
deltaMethod, 43
linearHypothesis, 80
∗Topic **package**
car-package, 4
∗Topic **print**
compareCoefs, 35
∗Topic **regression**
Anova, 7
avPlots, 13
bcPower, 17
Boot, 20
boxCox, 22
boxCoxVariable, 24
boxTidwell, 26
ceresPlots, 31
Contrasts, 36
crPlots, 39
deltaMethod, 43
dfbetaPlots, 49
durbinWatsonTest, 51
estimateTransform, 57
hccm, 65
hist.boot, 67
infIndexPlot, 70
influencePlot, 71
invResPlot, 72
invTranPlot, 74
leveragePlots, 78
linearHypothesis, 80
mmps, 89
ncvTest, 94
outlierTest, 97
plot.powerTransform, 100
powerTransform, 102
qqPlot, 105
residualPlots, 111
sigmaHat, 130
spreadLevelPlot, 134
subsets, 137
testTransform, 140
vif, 145
∗Topic **ts**
durbinWatsonTest, 51
∗Topic **univar**
qqPlot, 105
∗Topic **utilities**
showLabels, 128
some, 133
which.names, 149

abbreviate, *129*
abline, *111*
Adler, 5
AMSsurvey, 5
Angell, 6
Anova, 7, *84*
anova, *11*, *84*
anova.coxph, *11*
anova.glm, *11*
anova.lm, *11*
anova.mlm, *11*
Anscombe, 12
av.plot *(*car-deprecated*)*, 29
av.plots *(*car-deprecated*)*, 29
avp *(*avPlots*)*, 13
avPlot, *30*
avPlot *(*avPlots*)*, 13
avPlots, 13, *30*, *33*, *41*, *80*, *130*

basicPower, *143*
basicPower *(*bcPower*)*, 17
basicPowerAxis *(*TransformationAxes*)*, 142
Baumann, 16
bc *(*car-deprecated*)*, 29
bcPower, 17, *23*, *25*, *30*, *104*, *126*, *139, 140*, *143*
bcPowerAxis *(*TransformationAxes*)*, 142
Bfox, 18

Blackmoor, 19
Boot, 20, *70*
boot, *21*, *69*
boot.array, *21*
boot.ci, *21*
box.cox *(car-deprecated)*, 29
box.tidwell *(car-deprecated)*, 29
boxCox, 22
boxcox, *23*, *25*, *140*
boxCoxVariable, 24, *30*
Boxplot, 25, *121*, *123*
boxplot, *25, 26*, *123*, *140*
boxTidwell, 26, *30*
Burt, 28
bw.SJ, *47, 48*

CanPop, 28
car *(car-package)*, 4
car-deprecated, 29
car-package, 4
carWeb, 30
ceres.plot *(car-deprecated)*, 29
ceres.plots *(car-deprecated)*, 29
ceresPlot, *30*
ceresPlot *(ceresPlots)*, 31
ceresPlots, *15*, *30*, 31, *41*
Chile, 34
Chirot, 35
coef, *44*, *84*
compareCoefs, 35
confidence.ellipse *(car-deprecated)*, 29
confidenceEllipse, *30*
confidenceEllipse *(Ellipses)*, 53
confint.boot, *21*
confint.boot *(hist.boot)*, 67
contr.Helmert *(Contrasts)*, 36
contr.helmert, *38*
contr.poly, *38*
contr.Sum *(Contrasts)*, 36
contr.sum, *38*
contr.Treatment *(Contrasts)*, 36
contr.treatment, *38*
Contrasts, 36
cookd *(car-deprecated)*, 29
cooks.distance, *30*, *71, 72*
coplot, *99*
cov.trob, *54*, *56*, *123*, *126*
cov.wt, *54*, *56*
Cowles, 38
cr.plot *(car-deprecated)*, 29
cr.plots *(car-deprecated)*, 29
crossprod, *147, 148*
crp *(crPlots)*, 39
crPlot, *30*
crPlot *(crPlots)*, 39
crPlots, *15*, *30*, *33*, 39, *130*
cut, *109*

D, *46*
data.ellipse *(car-deprecated)*, 29
dataEllipse, *30*, *123*, *126*
dataEllipse *(Ellipses)*, 53
Davis, 41
DavisThin, 42
deltaMethod, 43
density, *46–48*, *70*
densityPlot, 46
Depredations, 48
dfbeta, *50*
dfbetaPlots, 49
dfbetas, *50*
dfbetasPlots *(dfbetaPlots)*, 49
Duncan, 51
durbin.watson *(car-deprecated)*, 29
durbinWatsonTest, *30*, 51
dwt *(durbinWatsonTest)*, 51

ellipse *(Ellipses)*, 53
Ellipses, 53
empinf, *21*
Ericksen, 56
estimateTransform, 57, *102–104*

factor, *109*
Florida, 59
Freedman, 60
Friendly, 61

gam, *91*, *113*, *118, 119*, *124*, *126, 127*
gamLine, *121*, *124*
gamLine *(ScatterplotSmoothers)*, 126
Ginzberg, 62
Greene, 62
Guyer, 63

Hartnagel, 64
hatvalues, *71, 72*
hccm, *9*, 65, *82–84*, *95*, *136*

head, *134*
Highway1, 66
hist, *70*
hist.boot, *21*, 67

identify, *114*, *129*
identify3d *(*scatter3d*)*, 116
infIndexPlot, 70
influence.plot *(*influencePlot*)*, 71
influenceIndexPlot *(*infIndexPlot*)*, 70
influencePlot, 71
inverseResponsePlot, *76*
inverseResponsePlot *(*invResPlot*)*, 72
invResPlot, 72
invTranEstimate *(*invTranPlot*)*, 74
invTranPlot, *73*, 74

jitter, *122, 123*

legend, *47*, *122, 123*
Leinhardt, 76
levene.test *(*car-deprecated*)*, 29
leveneTest, *30*, 77
leverage.plot *(*car-deprecated*)*, 29
leverage.plots *(*car-deprecated*)*, 29
leveragePlot, *30*
leveragePlot *(*leveragePlots*)*, 78
leveragePlots, *30*, 78, *130*
lht *(*linearHypothesis*)*, 80
linear.hypothesis *(*car-deprecated*)*, 29
linearHypothesis, *11*, *30*, *56*, 80
lines, *111*
lm, *26*, *102*, *114*, *139*
loess, *126, 127*
loessLine, *121*
loessLine *(*ScatterplotSmoothers*)*, 126
logit, 86, *143*

Mandel, 87
Manova *(*Anova*)*, 7
marginalModelPlot *(*mmps*)*, 89
marginalModelPlots *(*mmps*)*, 89
matchCoefs *(*linearHypothesis*)*, 80
Migration, 88
mmp *(*mmps*)*, 89
mmps, 89
Moore, 92
Mroz, 93

na.omit, *47*

ncv.test *(*car-deprecated*)*, 29
ncvTest, *30*, 94, *136*

OBrienKaiser, 95
optim, *58, 59*, *102*, *104*
optimize, *76*
Ornstein, 96
outlier.test *(*car-deprecated*)*, 29
outlierTest, *30*, *71*, 97

pairs, *126*
palette, *14*, *32*, *40*, *47*, *50*, *55*, *75*, *106*, *110*, *135*
panel.car, 99
par, *14*, *32, 33*, *40*, *55*, *79*, *106, 107*, *110*, *121, 122*, *125*, *135*, *138*, *143*
plot, *48*, *50*, *92*, *122*
plot.boot, *21*
plot.powerTransform, 100
points, *50*
Pottery, 101
powerTransform, *18*, *23*, *25*, *30*, *57–59*, *73*, *100*, 102, *125, 126*, *140, 141*
Prestige, 104
print.Anova.mlm *(*Anova*)*, 7
print.boxTidwell *(*boxTidwell*)*, 26
print.durbinWatsonTest *(*durbinWatsonTest*)*, 51
print.linearHypothesis.mlm *(*linearHypothesis*)*, 80
print.outlierTest *(*outlierTest*)*, 97
print.spreadLevelPlot *(*spreadLevelPlot*)*, 134
print.summary.Anova.mlm *(*Anova*)*, 7
printCoefmat, *36*
probabilityAxis, *87*
probabilityAxis *(*TransformationAxes*)*, 142

qq.plot *(*car-deprecated*)*, 29
qqline, *107*
qqnorm, *107*
qqp *(*qqPlot*)*, 105
qqPlot, *30*, 105
qqplot, *107*
qss, *127*
quantregLine *(*ScatterplotSmoothers*)*, 126
Quartet, 108

Recode *(*recode*)*, 108

recode, 108
regLine, *99*, 110
regsubsets, *137, 138*
regular expression, *83*
residCurvTest *(*residualPlots*)*, 111
residualPlot *(*residualPlots*)*, 111
residualPlots, *15*, 111, *130*
residuals.lm, *113*
Robey, 114
rqss, *126, 127*
rstudent, *71, 72*

Sahlins, 115
Salaries, 115
scatter3d, 116
scatterplot, 120, *126, 127*
scatterplot.matrix *(*car-deprecated*)*, 29
scatterplotMatrix, *30*, *123*, 123, *126, 127*
ScatterplotSmoothers, *32*, *40*, *90–92*, *113*, *121*, *123, 124*, *126*, 126
showLabels, *14*, *32, 33*, *40*, *50*, *55*, *70*, *72, 73*, *75*, *79*, *91*, *106, 107*, *113, 114*, *119*, *122–124*, *126*, 128
sigmaHat, 130
SLID, 131
slp *(*spreadLevelPlot*)*, 134
Soils, 132
some, 133
sp *(*scatterplot*)*, 120
spm *(*scatterplotMatrix*)*, 123
spread.level.plot *(*car-deprecated*)*, 29
spreadLevelPlot, *30*, *95*, 134
States, 136
subsets, 137
substr, *129*
summary.Anova.mlm *(*Anova*)*, 7
summary.boot, *21*
summary.boot *(*hist.boot*)*, 67
symbox, 139

tail, *134*
testTransform, *59*, *103, 104*, 140
Transact, 141
transform, *104*
TransformationAxes, 142
tukeyNonaddTest *(*residualPlots*)*, 111

UN, 144
USPop, 145

vcov, *44*, *84*
vcovHAC, *84*
vcovHC, *84*
vif, 145
Vocab, 147

waldtest, *84*
wcrossprod, 147
WeightLoss, 148
which.names, 149
whichNames *(*which.names*)*, 149
Womenlf, 150
Wool, 151

yjPower, *23*, *126*, *139, 140*, *143*
yjPower *(*bcPower*)*, 17
yjPowerAxis *(*TransformationAxes*)*, 142