Package 'effects'

August 14, 2014

Version 3.0-1

Date 2014/08/02

Title Effect Displays for Linear, Generalized Linear,Multinomial-Logit, Proportional-Odds Logit Models and
Mixed-Effects Models

Depends R (>= 2.10), lattice, grid, colorspace

Suggests nlme, lme4, MASS, nnet, poLCA, heplots

LazyLoad yes

LazyData yes

Description Graphical and tabular effect displays, e.g., of interactions, for various statistical models with linear predictors.

License GPL (>= 2)

URL http://www.r-project.org, http://socserv.socsci.mcmaster.ca/jfox/

Author John Fox [aut, cre],Sanford Weisberg [aut],Michael Friendly [aut],Jangman Hong [aut],Robert Andersen [ctb],David Firth [ctb],Steve Taylor [ctb]

Maintainer John Fox <jfox@mcmaster.ca>

Repository CRAN

Repository/R-Forge/Project effects

Repository/R-Forge/Revision 173

Repository/R-Forge/DateTimeStamp 2014-08-04 13:55:29

Date/Publication 2014-08-14 18:06:02

NeedsCompilation no

R topics documented:

`effects-package` *Effect Displays for Linear, Generalized Linear, Multinomial-Logit, Proportional-Odds Logit Models and Mixed-Effects Models*

Description

Graphical and tabular effect displays, e.g., of interactions, for linear (including fit via `gls`), multivariate-linear, generalized linear, multinomial-logit, proportional-odds logit, mixed-effect, polytomous latent-class, and some other models; (multidimensional) component+residual plots for linear and generalized linear models.

Details

Package:	effects
Version:	3.0-1
Date:	2014/08/02
Depends:	lattice, grid, colorspace
Suggests:	nlme, lme4, MASS, nnet, poLCA
LazyLoad:	yes
LazyData:	yes
License:	GPL (>= 2)
URL:	http://www.r-project.org, http://socserv.socsci.mcmaster.ca/jfox/

This package creates effect displays for various kinds of models, as partly explained in the references. Typical usage is `plot(allEffects(model))`, where `model` is an appropriate fitted-model object. Additional arguments to `allEffects` and `plot` can be used to customize the resulting displays. The function `effect` can be employed to produce an effect display for a particular term in the model, or to which terms in the model are marginal. The function `Effect` may similarly be used to produce an effect display for any combination of predictors. For linear and generalized linear

models it is also possible to plot partial residuals to obtain (multidimensional) component+residual plots. See `?effect`, `?Effect`, and `?plot.eff` for details.

Author(s)

John Fox <jfox@mcmaster.ca>, Sanford Weisberg, Michael Friendly, and Jangman Hong. We are grateful to Robert Andersen, David Firth, and for various suggestions.

Maintainer: John Fox <jfox@mcmaster.ca>

References

Fox, J. (1987) Effect displays for generalized linear models. *Sociological Methodology* **17**, 347–361.

Fox, J. (2003) Effect displays in R for generalised linear models. *Journal of Statistical Software* **8:15**, 1–27, <http://www.jstatsoft.org/counter.php?id=75&url=v08/i15/effect-displays-revised.pdf&ct=1>.

Fox, J. and R. Andersen (2006) Effect displays for multinomial and proportional-odds logit models. *Sociological Methodology* **36**, 225–255.

Fox, J. and J. Hong (2009). Effect displays in R for multinomial and proportional-odds logit models: Extensions to the effects package. *Journal of Statistical Software* **32:1**, 1–24, <http://www.jstatsoft.org/v32/i01/>.

`Arrests` *Arrests for Marijuana Possession*

Description

Data on police treatment of individuals arrested in Toronto for simple possession of small quantities of marijuana. The data are part of a larger data set featured in a series of articles in the Toronto Star newspaper.

Usage

```
Arrests
```

Format

A data frame with 5226 observations on the following 8 variables.

released Whether or not the arrestee was released with a summons; a factor with levels: `No`; `Yes`.

colour The arrestee's race; a factor with levels: `Black`; `White`.

year 1997 through 2002; a numeric vector.

age in years; a numeric vector.

sex a factor with levels: `Female`; `Male`.

employed a factor with levels: `No`; `Yes`.

citizen a factor with levels: `No`; `Yes`.

checks Number of police data bases (of previous arrests, previous convictions, parole status, etc. – 6 in all) on which the arrestee's name appeared; a numeric vector

Source

Personal communication from Michael Friendly, York University.

Examples

```
summary(Arrests)
```

`BEPS` *British Election Panel Study*

Description

These data are drawn from the 1997-2001 British Election Panel Study (BEPS).

Usage

```
BEPS
```

Format

A data frame with 1525 observations on the following 10 variables.

`vote` Party choice: `Conservative`, `Labour`, or `Liberal Democrat`

`age` in years

`economic.cond.national` Assessment of current national economic conditions, 1 to 5.

`economic.cond.household` Assessment of current household economic conditions, 1 to 5.

`Blair` Assessment of the Labour leader, 1 to 5.

`Hague` Assessment of the Conservative leader, 1 to 5.

`Kennedy` Assessment of the leader of the Liberal Democrats, 1 to 5.

`Europe` an 11-point scale that measures respondents' attitudes toward European integration. High scores represent 'Eurosceptic' sentiment.

`political.knowledge` Knowledge of parties' positions on European integration, 0 to 3.

`gender` `female` or `male`.

References

J. Fox and R. Andersen (2006) Effect displays for multinomial and proportional-odds logit models. *Sociological Methodology* **36**, 225–255.

Examples

```
summary(BEPS)


require(splines)  # for bs()
beps <- multinom(vote ~ age + gender + economic.cond.national + economic.cond.household
+ Blair + Hague + Kennedy + bs(Europe, 3)*political.knowledge, data=BEPS)
europe.knowledge <- effect("bs(Europe, 3)*political.knowledge", beps,
xlevels=list(Europe=seq(1, 11, length=50), political.knowledge=0:3),
   given.values=c(gendermale=0.5))
plot(europe.knowledge)
plot(europe.knowledge, style="stacked", colors=c("blue", "red", "orange"), rug=FALSE)
```

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 volunteering, 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.

Examples

```
summary(Cowles)
```

effect *Functions For Constructing Effect Plots*

Description

effect constructs an "eff" object for a term (usually a high-order term) in a linear (fit by lm or gls) or generalized linear model (fit by glm), or an "effpoly" object for a term in a multinomial or proportional-odds logit model (fit respectively by multinom or polr), absorbing the lower-order terms marginal to the term in question, and averaging over other terms in the model. For multivariate linear models (mlm), the function constructs a list of "eff" objects separately for the various response variables.

The function can also be used with mixed-effects models fit with lmer from the **lme4** package, or fit with lme from the **nlme** package, and for polytomous latent-class models fit by the poLCA function in the **poLCA** package. In mixed effects models the analysis is for the fixed effects only, not for random effects. The effect function works by constructing a call to Effect.

Effect also constructs an an "eff" object, but rather than focusing on a term in the model, it focuses on a subset of the predictors. Effect is consequently more flexible and robust than effect, and will work with some models for which effect fails, such as models with nested terms (see the examples). The effect function calls Effect. There is a default method for Effect that should work with any model object that has a linear predictor and responds to the coef, model.frame, formula, and vcov functions.

allEffects identifies all of the high-order terms in a model and returns a list of "eff" or "effpoly" objects (i.e., an object of type "efflist").

Usage

```
effect(term, mod, vcov.=vcov, ...)

Effect(focal.predictors, mod, ...)

## S3 method for class 'lm'
Effect(focal.predictors, mod, xlevels = list(), default.levels = NULL, given.values,
    vcov.=vcov, se = TRUE, confidence.level = 0.95,
   transformation = list(link = family(mod)$linkfun, inverse = family(mod)$linkinv),
    typical = mean, offset = mean,
    partial.residuals=FALSE, quantiles=seq(0.2, 0.8, by=0.2),
    x.var=NULL,  ...)

## S3 method for class 'gls'
Effect(focal.predictors, mod, xlevels = list(), default.levels=NULL,
   given.values, vcov.=vcov, se = TRUE, confidence.level = 0.95, transformation = NULL,
    typical = mean, ...)

## S3 method for class 'multinom'
Effect(focal.predictors, mod,
    confidence.level=.95, xlevels=list(), default.levels=NULL,
```

```
    given.values, vcov.=vcov, se=TRUE, typical=mean, ...)

## S3 method for class 'polr'
Effect(focal.predictors, mod,
    confidence.level=.95, xlevels=list(), default.levels=NULL,
    given.values, vcov.=vcov, se=TRUE, typical=mean, latent=FALSE, ...)

## S3 method for class 'mer'
Effect(focal.predictors, mod, ...)

## S3 method for class 'merMod'
Effect(focal.predictors, mod, ...)

## S3 method for class 'lme'
Effect(focal.predictors, mod, ...)

## S3 method for class 'poLCA'
Effect(focal.predictors, mod, ...)

## S3 method for class 'mlm'
Effect(focal.predictors, mod, response, ...)

## Default S3 method:
Effect(focal.predictors, mod, xlevels = list(),
    default.levels = NULL, given.values,
    vcov. = vcov, se = TRUE, confidence.level = 0.95,
    transformation = list(link = I, inverse = I),
    typical = mean, offset = mean, ...)

allEffects(mod, ...)

## S3 method for class 'mer'
allEffects(mod, ...)

## S3 method for class 'merMod'
allEffects(mod, ...)

## S3 method for class 'lme'
allEffects(mod, ...)

## S3 method for class 'poLCA'
allEffects(mod, ...)

## S3 method for class 'eff'
as.data.frame(x, row.names=NULL, optional=TRUE,
    transform=x$transformation$inverse, ...)

## S3 method for class 'effpoly'
```

```
as.data.frame(x, row.names=NULL, optional=TRUE, ...)

## S3 method for class 'efflatent'
as.data.frame(x, row.names=NULL, optional=TRUE, ...)

## S3 method for class 'eff'
vcov(object, ...)
```

Arguments

term	the quoted name of a term, usually, but not necessarily, a high-order term in the model. The term must be given exactly as it appears in the printed model, although either colons (:) or asterisks (*) may be used for interactions.
focal.predictors	a character vector of one or more predictors in the model.
mod	an object of class "lm", "gls", "glm", "multinom", "polr", "mer" (or "merMod"), "lme" or "poLCA".
xlevels	this argument is used to set the number of levels for any focal predictor that is not a factor. If xlevels=NULL, the default, then the number and values of levels for any numeric predictor is determined by grid.pretty. If xlevels=n is an integer, then each numeric predictor is represented by n equally spaced levels. More generally, xlevels can be a named list of values at which to set each numeric predictor. For example, xlevels=list(x1=c(2, 4, 7), x2=5) would use the values 2, 4 and 7 for the levels of x1, 5 equally spaced levels for the levels of x2, and use the default for any other numeric predictors. If partial residuals are computed, then the focal predictor that is to appear on the horizontal axis of an effect plot is evaluated at 100 equally spaced values along its full range, and, by default, other numeric predictors are evaluated at the quantiles specified in the quantiles argument, unless their values are given explicitly in xlevels.
default.levels	ignored, but included for compatibility with pre-July 2013 versions of this package. Use xlevels instead.
given.values	a numeric vector of named elements, setting particular columns of the model matrix to specific values for predictors that are *not* focal predictors; if specified, this argument takes precedence over the application of the function given in the typical argument (below). Care must be taken in specifying these values — e.g., for a factor, the values of all contrasts should be given and these should be consistent with each other.
vcov.	A function or the name of a function that will be used to get the estimated variance covariance matrix of the estimated coefficients. This will ordinarily be the default, vcov, which will result in the function call vcov(mod) to get the variance covariance matrix. You can use the name of any function with mod as the value of its first argument that returns an estimated sample covariance matrix, such as the hccm function in the **car** package that returns a heteroscedasticity corrected estimate with linear models.
se	if TRUE, the default, calculate standard errors and confidence limits for the effects. For mer and lme objects, the normal distribution is used to get confidence limits.

confidence.level
: level at which to compute confidence limits based on the standard-normal distribution; the default is 0.95.

transformation
: a two-element list with elements link and inverse. For a generalized linear model, these are by default the link function and inverse-link (mean) function. For a linear model, these default to NULL. If NULL, the identify function, I, is used; this effect can also be achieved by setting the argument to NULL. The inverse-link may be used to transform effects when they are printed or plotted; the link may be used in positioning axis labels (see below). If the link is not given, an attempt will be made to approximate it from the inverse-link.

typical
: a function to be applied to the columns of the model matrix over which the effect is "averaged"; the default is mean.

offset
: a function to be applied to the offset values (if there is an offset) in a linear or generalized linear model, or a mixed-effects model fit by lmer or glmer; or a numeric value, to which the offset will be set. The default is the mean function, and thus the offset will be set to its mean. *Note:* Only offsets defined by the offset argument to lm, glm, lmer, or glmer will be handled correctly; use of the offset function in the model formula is not supported.

partial.residuals
: if TRUE, partial residuals will be computed for an effect in a linear or generalized linear model; if FALSE (the default), partial residuals are suppressed.

quantiles
: quantiles at which to evaluate numeric focal predictors *not* on the horizontal axis, used only when partial residuals are displayed; superceded if the xlevels argument gives specific values for a predictor.

x.var
: the name or index of the numeric predictor to define the horizontal axis of an effect plot for a linear or generalized linear model; the default is NULL, in which case the first numeric predictor in the effect will be used *if* partial residuals are to be computed. This argument is intended to be used when partial.residuals is TRUE; otherwise, the variable on the horizontal axis can be chosen when the effect object is plotted: see plot.eff.

latent
: if TRUE, effects in a proportional-odds logit model are computed on the scale of the latent response; if FALSE (the default) effects are computed as individual-level probabilities and logits.

x
: an object of class "eff", "effpoly", or "efflatent".

transform
: a transformation to be applied to the effects and confidence limits, by default taken from the inverse link function saved in the "eff" object.

row.names, optional
: not used.

response
: for an mlm, a vector containing the name(s) or indices of one or more response variable(s). The default is to use all responses in the model.

object
: an object of class "eff" for which the covariance matrix of the effects is desired.

...
: arguments to be passed down.

Details

Normally, the functions to be used directly are `allEffects`, to return a list of high-order effects, and the generic `plot` function to plot the effects. (see `plot.efflist`, `plot.eff`, and `plot.effpoly`). Alternatively, `Effect` can be used to vary a subset of predictors over their ranges, while other predictors are held to typical values. Plots are drawn using the `xyplot` (or in some cases, the `densityplot`) function in the `lattice` package. Effects may also be printed (implicitly or explicitly via `print`) or summarized (using `summary`) (see `print.efflist`, `summary.efflist`, `print.eff`, `summary.eff`, `print.effpoly`, and `summary.effpoly`).

If asked, the `effect` function will compute effects for terms that have higher-order relatives in the model, averaging over those terms (which rarely makes sense), or for terms that do not appear in the model but are higher-order relatives of terms that do. For example, for the model `Y ~ A*B + A*C + B*C`, one could compute the effect corresponding to the absent term `A:B:C`, which absorbs the constant, the `A`, `B`, and `C` main effects, and the three two-way interactions. In either of these cases, a warning is printed.

The `as.data.frame` methods convert effect objects to data frames to facilitate the construction of custom displays. In the case of `"eff"` objects, the `se` element in the data frame is always on the scale of the linear predictor, and the transformation used for the fit and confidence limits is saved in a `"transformation"` attribute.

Value

For `lm`, `glm`, `mer` and `lme`, `effect` and `Effect` return an `"eff"` object, and for `multinom` and `polr`, an `"effpoly"` object, with the components listed below. For `mlm` with one response specified, an `"eff"` object, otherwise a class `"efflist"` object, containing one `"eff"` object for each `response`.

`term`	the term to which the effect pertains.
`formula`	the complete model formula.
`response`	a character string giving the name of the response variable.
`y.levels`	(for `"effpoly"` objects) levels of the polytomous response variable.
`variables`	a list with information about each predictor, including its name, whether it is a factor, and its levels or values.
`fit`	(for `"eff"` objects) a one-column matrix of fitted values, representing the effect on the scale of the linear predictor; this is a ravelled table, representing all combinations of predictor values.
`prob`	(for `"effpoly"` objects) a matrix giving fitted probabilities for the effect for the various levels of the the response (columns) and combinations of the focal predictors (rows).
`logit`	(for `"effpoly"` objects) a matrix giving fitted logits for the effect for the various levels of the the response (columns) and combinations of the focal predictors (rows).
`x`	a data frame, the columns of which are the predictors in the effect, and the rows of which give all combinations of values of these predictors.
`model.matrix`	the model matrix from which the effect was calculated.
`data`	a data frame with the data on which the fitted model was based.

discrepancy — the percentage discrepancy for the 'safe' predictions of the original fit; should be very close to 0. Note: except for gls models, this is now necessarily 0.

offset — value to which the offset is fixed; 0 if there is no offset.

model — (for "effpoly" objects) "multinom" or "polr", as appropriate.

vcov — (for "eff" objects) a covariance matrix for the effect, on the scale of the linear predictor.

se — (for "eff" objects) a vector of standard errors for the effect, on the scale of the linear predictor.

se.prob, se.logit — (for "effpoly" objects) matrices of standard errors for the effect, on the probability and logit scales.

lower, upper — (for "eff" objects) one-column matrices of confidence limits, on the scale of the linear predictor.

lower.prob, upper.prob, lower.logit, upper.logit — (for "effpoly" objects) matrices of confidence limits for the fitted logits and probabilities; the latter are computed by transforming the former.

confidence.level — for the confidence limits.

transformation — (for "eff" objects) a two-element list, with element link giving the link function, and element inverse giving the inverse-link (mean) function.

fitted.rounded — partial fitted values at the observed values of the predictor x.var and the values of the other predictors that appear in the effect display; predictors not in the effect are held constant to typical values. This and the following two elements are NULL if partial residuals aren't computed and pertain only to linear or generalized linear models.

fitted — partial fitted values for the observed values of all predictors that appear in the effect display; predictors not in the effect are held constant to typical values.

partial.residuals.raw — partial residuals for the effect computed at the actual values of all focal predictors.

partial.residuals.adjusted — partial residuals for the effect computed at the panel-rounded values of focal predictors, except for the predictor corresponding to xvar, which will appear on the horizontal axis of the plotted effect.

x.var — the name of the predictor to appear on the horizontal axis of an effect plot made from the returned object; will usually be NULL if partial residuals aren't computed.

effectList returns a list of "eff" or "effpoly" objects corresponding to the high-order terms of the model.

If mod is of class poLCA (from the poLCA package) to fit a polytomous latent class model, effects are computed for the predictors given the estimated latent classes. The result is of class eff if the latent class model has 2 categories and of class effpoly with more than 2 categories.

Warnings and Limitations

The `Effect` function handles factors and covariates differently, and is likely to be confused if one is changed to the other in a model formula. Consequently, formulas that include calls to `as.factor`, `factor`, or `numeric` (as, e.g., in `y ~ as.factor(income)`) will cause errors. Instead, create the modified variables outside of the model formula (e.g., `fincome <- as.factor(income)`) and use these in the model formula. Similarly variables of class `date` or `"times"`, which are usually differences between `"dates"` variables, should be converted to numeric variables outside the model formula.

Factors cannot have colons in level names (e.g., `"level:A"`); the `effect` function will confuse the colons with interactions; rename levels to remove or replace the colons (e.g., `"level.A"`). In addition, factors cannont be declared on the fly (e.g., using `y ~ a + factor(b)`.

The functions in the **effects** package work properly with predictors that are numeric or factors; consequently, e.g., convert logical predictors to factors, and dates to numeric.

Empty cells in crossed-factors are now permitted for lm, glm and multinom models.

With multinom models with two or more crossed factors with an empty cell, the 'plot' command with `style="stacked"` apparently does not work because of a bug in the `barchart` function in `lattice`. However, the default `style="lines"` does work.

Offsets in linear and generalized linear models are supported, as are offsets in mixed models fit by `lmer` or `glmer`, but must be supplied through the `offset` argument to `lm`, `glm`, `lmer` or `glmer`; offsets supplied via calls to the `offset` function on the right-hand side of the model formula are not supported.

Calling any of these functions from within a user-written function may result in errors due to R's scoping rules. See the vignette `embedding.pdf` for the **car** package for a solution to this problem.

Author(s)

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

References

Fox, J. (1987). Effect displays for generalized linear models. *Sociological Methodology* **17**, 347–361.

Fox, J. (2003) Effect displays in R for generalised linear models. *Journal of Statistical Software* **8:15**, 1–27, <http://www.jstatsoft.org/v08/i15/>.

Fox, J. and R. Andersen (2006). Effect displays for multinomial and proportional-odds logit models. *Sociological Methodology* **36**, 225–255.

Fox, J. and J. Hong (2009). Effect displays in R for multinomial and proportional-odds logit models: Extensions to the effects package. *Journal of Statistical Software* **32:1**, 1–24, <http://www.jstatsoft.org/v32/i01/>.

Hastie, T. J. (1992). Generalized additive models. In Chambers, J. M., and Hastie, T. J. (eds.) *Statistical Models in S*, Wadsworth.

Weisberg, S. (2014). *Applied Linear Regression*, 4th edition, Wiley, http://z.umn.edu/alr4ed.

See Also

print.eff, summary.eff, plot.eff, print.summary.eff, print.effpoly, summary.effpoly, plot.effpoly, print.efflist, summary.efflist, plot.efflist, xyplot, densityplot

Examples

```
# Note: Some of these examples are marked as "don't run"
#       to reduce the execution times of the examples.

mod.cowles <- glm(volunteer ~ sex + neuroticism*extraversion,
    data=Cowles, family=binomial)
eff.cowles <- allEffects(mod.cowles, xlevels=list(extraversion=seq(0, 24, 6)),
    given.values=c(sexmale=0.5))
eff.cowles
as.data.frame(eff.cowles[[2]])

## Not run:
# the following are equivalent:
eff.ne <- effect("neuroticism*extraversion", mod.cowles)
Eff.ne <- Effect(c("neuroticism", "extraversion"), mod.cowles)
all.equal(eff.ne$fit, Eff.ne$fit)

plot(eff.cowles, 'sex', ylab="Prob(Volunteer)")

plot(eff.cowles, 'neuroticism:extraversion', ylab="Prob(Volunteer)",
    ticks=list(at=c(.1,.25,.5,.75,.9)))

plot(eff.cowles, 'neuroticism:extraversion', multiline=TRUE,
    ylab="Prob(Volunteer)")

plot(effect('sex:neuroticism:extraversion', mod.cowles,
    xlevels=list(extraversion=seq(0, 24, 6))), multiline=TRUE)

## End(Not run)

# a nested model:

mod <- lm(log(prestige) ~ income:type + education, data=Prestige)

# does not work: effect("income:type", mod, transformation=list(link=log, inverse=exp))

plot(Effect(c("income", "type"), mod, transformation=list(link=log, inverse=exp),
ylab="prestige")) # works


if (require(nnet)){
mod.beps <- multinom(vote ~ age + gender + economic.cond.national +
economic.cond.household + Blair + Hague + Kennedy +
Europe*political.knowledge, data=BEPS)
## Not run:
plot(effect("Europe*political.knowledge", mod.beps,
xlevels=list(political.knowledge=0:3)))
```

```
## End(Not run)

plot(Effect(c("Europe", "political.knowledge"), mod.beps,
        xlevels=list(Europe=1:11, political.knowledge=0:3),
given.values=c(gendermale=0.5)),
style="stacked", colors=c("blue", "red", "orange"), rug=FALSE)

## Not run:
plot(effect("Europe*political.knowledge", mod.beps, # equivalent
     xlevels=list(political.knowledge=0:3),
given.values=c(gendermale=0.5)),
style="stacked", colors=c("blue", "red", "orange"), rug=FALSE)

## End(Not run)
}

if (require(MASS)){
mod.wvs <- polr(poverty ~ gender + religion + degree + country*poly(age,3),
data=WVS)
## Not run:
plot(effect("country*poly(age, 3)", mod.wvs))

## End(Not run)

plot(Effect(c("country", "age"), mod.wvs), style="stacked")

## Not run:
plot(effect("country*poly(age, 3)", mod.wvs), style="stacked") # equivalent

plot(effect("country*poly(age, 3)", latent=TRUE, mod.wvs))

## End(Not run)
}


mod.pres <- lm(prestige ~ log(income, 10) + poly(education, 3) + poly(women, 2),
    data=Prestige)
eff.pres <- allEffects(mod.pres, xlevels=50)
plot(eff.pres)
plot(eff.pres[1],
    transform.x=list(income=list(trans=log10, inverse=function(x) 10^x)),
    ticks.x=list(income=list(at=c(1000, 2000, 5000, 10000, 20000))))
## Not run:
# linear model with log-response and log-predictor
# to illustrate transforming axes and setting tick labels
mod.pres1 <- lm(log(prestige) ~ log(income) + poly(education, 3) + poly(women, 2),
    data=Prestige)
# effect of the log-predictor
eff.log <- Effect(c("income"), mod.pres1)
# effect of the log-predictor transformed to the arithmetic scale
eff.trans <- Effect(c("income"), mod.pres1, transformation=list(link=log, inverse=exp))
#variations:
```

```
# y-axis:  scale is log, tick labels are log
# x-axis:  scale is arithmetic, tick labels are arithmetic
plot(eff.log)

# y-axis:  scale is log, tick labels are log
# x-axis:  scale is log, tick labels are arithmetic
plot(eff.log, transform.x=list(income=c(trans=log, inverse=exp)),
       ticks.x=list(income=list(at=c(1000, 2000, 5000, 10000, 20000))),
       xlab="income, log-scale")

# y-axis:  scale is log, tick labels are airthmetic
# x-axis:  scale is arithmetic, tick labels are arithmetic
plot(eff.trans, ylab="prestige")

# y-axis:  scale is arithmetic, tick labels are airthmetic
# x-axis:  scale is arithmetic, tick labels are arithmetic
plot(eff.trans, rescale.axis=FALSE, ylab="prestige")

# y-axis:  scale is log, tick labels are arithmetic
# x-axis:  scale is log, tick labels are arithmetic
plot(eff.trans, transform.x=list(income=c(trans=log, inverse=exp)),
       ticks.x=list(income=list(at=c(1000, 2000, 5000, 10000, 20000))),
       xlab="income, log-scale", ylab="prestige, log-scale",
       main="Both effect and X in log-scale")

# y-axis:  scale is arithmetic, tick labels are airthmetic
# x-axis:  scale is log, tick labels are arithmetic
plot(eff.trans, transform.x=list(income=c(trans=log, inverse=exp)),
       ticks.x=list(income=list(at=c(1000, 2000, 5000, 10000, 20000))),
       rescale.axis=FALSE,
       xlab="income, log-scale", ylab="prestige")

## End(Not run)

if (require(nlme)){ # for gls()
mod.hart <- gls(fconvict ~ mconvict + tfr + partic + degrees, data=Hartnagel,
    correlation=corARMA(p=2, q=0), method="ML")
plot(allEffects(mod.hart))
    detach(package:nlme)
}

if (require(lme4)){
data(cake, package="lme4")
fm1 <- lmer(angle ~ recipe * temperature + (1|recipe:replicate), cake,
              REML = FALSE)
plot(Effect(c("recipe", "temperature"), fm1))
## Not run:
plot(effect("recipe:temperature", fm1), grid=TRUE) # equivalent

## End(Not run)
detach(package:lme4)
}
```

```
## Not run:
if (require(nlme) && length(find.package("lme4", quiet=TRUE)) > 0){
data(cake, package="lme4")
cake$rep <- with(cake, paste( as.character(recipe), as.character(replicate), sep=""))
fm2 <- lme(angle ~ recipe * temperature, data=cake,
           random = ~ 1 | rep, method="ML")
plot(Effect(c("recipe", "temperature"), fm2))
plot(effect("recipe:temperature", fm2), grid=TRUE)  # equivalent
}
detach(package:nlme)

## End(Not run)

## Not run:
if (require(poLCA)){
  data(election)
  f2a <- cbind(MORALG,CARESG,KNOWG,LEADG,DISHONG,INTELG,
              MORALB,CARESB,KNOWB,LEADB,DISHONB,INTELB)~PARTY*AGE
  nes2a <- poLCA(f2a,election,nclass=3,nrep=5)
  plot(Effect(c("PARTY", "AGE"), nes2a), style="stacked")
}

## End(Not run)

# mlm example
if (require(heplots)) {
data(NLSY, package="heplots")
mod <- lm(cbind(read,math) ~ income+educ, data=NLSY)
eff.inc <- Effect("income", mod)
plot(eff.inc)
eff.edu <- Effect("educ", mod)
plot(eff.edu, rug=FALSE, grid=TRUE)
    ## Not run:
plot(Effect("educ", mod, response="read"))

## End(Not run)
  detach(package:heplots)
}

# component + residual plot examples

## Not run:
Prestige$type <- factor(Prestige$type, levels=c("bc", "wc", "prof"))

mod.prestige.1 <- lm(prestige ~ income + education, data=Prestige)
plot(allEffects(mod.prestige.1, partial.residuals=TRUE)) # standard C+R plots

mod.prestige.2 <- lm(prestige ~ type*(income + education), data=Prestige)
plot(allEffects(mod.prestige.2, partial.residuals=TRUE))

mod.prestige.3 <- lm(prestige ~ type + income*education, data=Prestige)
plot(Effect(c("income", "education"), mod.prestige.3, partial.residuals=TRUE),
  span=1)
```

```
## End(Not run)

  #  artificial data

set.seed(12345)
x1 <- runif(500, -75, 100)
x2 <- runif(500, -75, 100)
y <- 10 + 5*x1 + 5*x2 + x1^2 + x2^2 + x1*x2 + rnorm(500, 0, 1e3)
Data <- data.frame(y, x1, x2)
mod.1 <- lm(y ~ poly(x1, x2, degree=2, raw=TRUE), data=Data)
                        # raw=TRUE necessary for safe prediction
mod.2 <- lm(y ~ x1*x2, data=Data)
mod.3 <- lm(y ~ x1 + x2, data=Data)

plot(Effect(c("x1", "x2"), mod.1, partial.residuals=TRUE)) # correct model
plot(Effect(c("x1", "x2"), mod.2, partial.residuals=TRUE)) # wrong model
plot(Effect(c("x1", "x2"), mod.3, partial.residuals=TRUE)) # wrong model
```

`effects-deprecated` *Deprecated Function in effects Package*

Description

The `all.effects` function is provided only for compatibility with older versions of the effects package and may be removed; use `allEffects` instead.

Usage

```
all.effects(...)
```

Arguments

`...`	arguments to be passed to `allEffects`.

Author(s)

John Fox <jfox@mcmaster.ca>.

See Also

`allEffects`

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.

`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 (years) of occupational incumbents, in 1971.

income Average income (dollars) of incumbents, 1971.

women Percentage of incumbents who are women, 1971.

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. (1997) *Applied Regression, Linear Models, and Related Methods.* Sage.

summary.eff *Summarizing, Printing, and Plotting Effects*

Description

summary, print, plot, and [methods for eff, effpoly, efflist, and mlm.efflist objects.

Usage

```
## S3 method for class 'eff'
print(x, type=c("response", "link"), ...)
## S3 method for class 'effpoly'
print(x, type=c("probability", "logits"), ...)
## S3 method for class 'efflatent'
print(x, ...)
## S3 method for class 'efflist'
print(x, ...)
## S3 method for class 'mlm.efflist'
print(x, ...)
## S3 method for class 'summary.eff'
print(x, ...)
## S3 method for class 'eff'
summary(object, type=c("response", "link"), ...)
## S3 method for class 'effpoly'
summary(object, type=c("probability", "logits"), ...)
## S3 method for class 'efflatent'
summary(object, ...)
## S3 method for class 'efflist'
summary(object, ...)
## S3 method for class 'mlm.efflist'
summary(object, ...)
## S3 method for class 'eff'
plot(x, x.var,
    z.var=which.min(levels), multiline=is.null(x$se), rug=TRUE,
    xlab, ylab, main=paste(effect, "effect plot"),
    colors=palette(), symbols=1:length(colors), lines=1:length(colors),
    cex=1.5, lwd=2, ylim, xlim=NULL,
    factor.names=TRUE, ci.style, band.transparency=0.15, band.colors=colors,
    type=c("response", "link"), ticks=list(at=NULL, n=5),
    alternating=TRUE, rotx=0, roty=0, grid=FALSE, layout, rescale.axis=TRUE,
    transform.x=NULL, ticks.x=NULL,
    key.args=NULL,
    row=1, col=1, nrow=1, ncol=1, more=FALSE,
   use.splines=TRUE, partial.residuals=c("adjusted", "raw"), show.fitted=FALSE,
    residuals.color="blue", residuals.pch=1, span=2/3, ...)
## S3 method for class 'effpoly'
plot(x,
```

```
type=c("probability", "logit"),
x.var=which.max(levels),
rug=TRUE,
xlab,
ylab=paste(x$response, " (", type, ")", sep=""),
main=paste(effect, "effect plot"),
colors, symbols, lines, cex=1.5, lwd=2,
factor.names=TRUE, ci.style, band.colors, band.transparency=0.3,
style=c("lines", "stacked"),
confint=(style == "lines" && !is.null(x$confidence.level)),
transform.x=NULL, ticks.x=NULL, xlim=NULL,
ylim, rotx=0, alternating=TRUE, roty=0, grid=FALSE,
layout, key.args=NULL,
row=1, col=1, nrow=1, ncol=1, more=FALSE, use.splines=TRUE, ...)
## S3 method for class 'efflist'
plot(x, selection, rows, cols, ask=FALSE, graphics=TRUE, ...)
## S3 method for class 'mlm.efflist'
plot(x, ...)
## S3 method for class 'efflist'
x[...]
```

Arguments

x	an object of class "eff", "effpoly", "efflist", "mlm.efflist", or "summary.eff", as appropriate.
object	an object of class "eff", "effpoly", "efflist", or "mlm.efflist", as appropriate.
type	for linear and generalized linear models, if "response" (the default), effects are printed or the vertical axis is labelled on the scale of the response variable; if "link", effects are printed or the vertical axis labelled on the scale of the linear predictor. For polytomous logit models, this argument takes either "probability" or "logit", with the former as the default.
x.var	the index (number) or quoted name of the covariate or factor to place on the horizontal axis of each panel of the effect plot. The default is the predictor with the largest number of levels or values.
z.var	for linear, generalized linear or mixed models, the index (number) or quoted name of the covariate or factor for which individual lines are to be drawn in each panel of the effect plot. The default is the predictor with the smallest number of levels or values. This argument is only used if multiline = TRUE.
multiline	for linear, generalized linear or mixed models, if TRUE, each panel of the display represents combinations of values of two predictors, with one predictor (corresponding to x.var) on the horzontal axis, and the other (corresponding to z.var) used to define lines in the graph; defaults to TRUE if there are no standard errors in the object being plotted, and FALSE otherwise.
confint	plot point-wise confidence bands around fitted effects (for multinomial and proportional-odds logit models); defaults to TRUE, in which case separate panels are used for different response levels.

`rug` if `TRUE`, the default, a rug plot is shown giving the *marginal* distribution of the predictor on the horizontal axis, if this predictor is a covariate. The rug plot is suppressed if partial residuals are plotted.

`xlab` the label for the horizontal axis of the effect plot; if missing, the function will use the name of the predictor on the horizontal axis.

`ylab` the label for the vertical axis of the effect plot; the default is constructed from the name of the response variable for the model from which the effect was computed.

`main` the title for the plot, printed at the top; the default title is constructed from the name of the effect.

`colors` `colors[1]` is used to plot effects, `colors[2]` to plot confidence limits when `ci.style` is not equal to `"bands"`. In a mulitline plot, the successive `colors` correspond to the levels of the `z.var` covariate or factor. In a stacked plot or a plot without confidence bands for a multinomial or proportional-odds logit model, the successive `colors` correspond to the levels of the response factor. In all but stacked plots, `colors` defaults to `palette()`. If `colors` is shorter than the number of levels, then it is recycled, so `colors="black"` will use black for all levels.

For stacked multinomial-logit plots, `colors` defaults to `rainbow_hcl(levels)`, where `levels` is the number of levels of the response variable; for stacked proportional-odds model plots, `colors` defautls to `sequential_hcl(levels)`. `colors` does not recycle for stacked plots.

Warning: This argument *cannot* be abbreviated to `col`, which is used for a different purpose (see below).

`symbols, lines` corresponding to the levels of the `z.var` covariate or factor on a multiline plot, or to the successive levels of the response factor in a line plot for a polytomous logit model. These arguments are used only if `multiline = TRUE` or for polytomous logit models where the effects are plotted without confidence bands; in these cases a legend is drawn at the top of the display. If these arguments are too short they are recycled.

`cex` character expansion for plotted symbols; default is `1.5`.

`lwd` line width for fitted lines

`ylim` 2-element vector containing the lower and upper limits of the vertical axes; if `NULL`, the default, then the vertical axes are scaled from the data.

`xlim` a named list of 2-element vectors, with the names corresponding to numeric predictors; if a numeric predictor is in the list, then when it appears on the horizontal axis, the axis limits will be taken from the corresponding vector; if a predictor is not in the list, or if the argument is `NULL` (the default), then the horizontal axis limits are computed from the data.

`factor.names` a logical value, default `TRUE`, that controls the inclusion of factor names in conditioning-variable labels.

`ci.style` confidence bounds can be indicated using error bars, using lines or confidence bands, depending on the plot type. For single line plots the default is `"bars"` for factors and `"bands"` for variates. Style `"lines"` can also be used for either of these. For multiline plots, the default is `"none"` for no confidence bounds,

but style "bars" or "bands" can also be used. For a variate the option "bars" displays the error bars at each of the levels points for which the horizontal variable was evaluated.

band.colors A vector of colors for the color of the confidence band with ci.style="bands". The default is band.colors=colors. For plots with one line, the choice, setting band.colors="red" produces a pleasing result, even if color provides no additional information in this case. If this argument is too short it is recycled.

band.transparency
For ci.style="bands", the alpha transparency of the fill color. Not all graphic devices support transparency.

style (for multinomial or proportional-odds logit models) "lines" (the default for a line plot, or "stacked" for a stacked-bar or stacked-area plot. In the latter case only fitted probabilities may be plotted and confidence envelopes cannot be shown.

ticks a two-item list controlling the placement of tick marks on the vertical axis, with elements at and n. If at=NULL (the default), the program attempts to find 'nice' locations for the ticks, and the value of n (default, 5) gives the approximate number of tick marks desired; if at is non-NULL, then the value of n is ignored.

ticks.x a named list of two-item lists controlling the placement of tick marks on the horizontal axis. Each list element is named for a numeric predictor in the model, and each sublist has elements at or n are for the ticks argument. If a predictor doesn't appear in the list, or if ticks.x is NULL (the default), then the tick marks are computed by the function.

transform.x transformations to be applied to the horizontal axis, in the form of a named list, each of whose elements is itself a list of two functions, with sublist element names trans and inverse. The names of the list elements are numeric predictors in the model whose axes are to be transformed; the trans function is applied to the values of the predictor, and inverse is used for computing proper axis tick labels. If a numeric predictor is missing from transform.x then its axis is not transformed; if the argument is NULL (the default), then no predictor axes are transformed.

alternating if TRUE (the default), the tick labels alternate by panels in multi-panel displays from left to right and top to bottom; if FALSE, tick labels appear at the bottom and on the left.

rotx, roty rotation angles for the horizontal and vertical tick marks, respectively. Default is 0.

grid if TRUE, add grid lines to the plot. Default is FALSE.

layout the layout argument to the **lattice** function xyplot (or, in some cases densityplot), which is used to draw the effect display; if not specified, the plot will be formatted so that it appears on a single page.

rescale.axis if TRUE (the default), the tick marks on the vertical axis are labelled on the response scale (e.g., the probability scale for effects computed on the logit scale for a binomial GLM).

key.args additional arguments to be passed to the key trellis argument to xyplot or densityplot, e.g., to position the key (legend) in the plotting region.

`row, col, nrow, ncol, more`
: These arguments are used to graph an effect as part of an array of plots; `row`, `col`, `nrow`, and `ncol` are used to compose the `split` argument and `more` the `more` argument to `print.trellis`. Normally these arguments are not set by the user, but by `plot.efflist`. **Warning:** Note that `col` is *not* used to specify colors; use `colors` instead (see above).

`selection`
: the optional index (number) or quoted name of the effect in an effect list to be plotted; if not supplied, a menu of high-order terms is presented or all effects are plotted.

`rows, cols`
: Number of rows and columns in the "meta-array" of plots produced for an `efflist` object; if either argument is missing, then the meta-layout will be computed by the `plot` method.

`ask`
: if `selection` is not supplied and `ask` is `TRUE`, a menu of high-order terms is presented; if `ask` is `FALSE` (the default), effects for all high-order terms are plotted in an array.

`graphics`
: if `TRUE` (the default), then the menu of terms to plot is presented in a dialog box rather than as a text menu.

`use.splines`
: If `TRUE`, the default, then any lines drawn when the horizontal axis is not a factor use interpolating splines computed by the `spline` function. If FALSE, then linear interpoliation is used. This argument is ignored if the horizontal axis is a factor.

`partial.residuals`
: use `"adjusted"` partial residuals, computed at the panel-rounded values of the focal predictors not on the horizontal axis of each panel, or `"raw"` partial residuals, computed at the actual values of all focal predictors; the default is `"adjusted"`, and this argument is effective only when partial residuals are included in the effect object to be plotted — see `Effect`.

`span`
: of the *loess* smoother to be applied to partial residuals, if present; the default is `2/3`.

`show.fitted`
: if partial residuals are present in the effect object, also plot the partial fitted values (which will be shown as filled circles).

`residuals.color`
: color for plotting partial residuals (default `"blue"`).

`residuals.pch`
: plotting symbol for partial residuals (default `1`, open circles).

`...`
: arguments to be passed down.

Details

In a generalized linear model, by default, the `print` and `summary` methods for `eff` objects print the computed effects on the scale of the response variable using the inverse of the link function. In a logit model, for example, this means that the effects are expressed on the probability scale.

By default, effects in a GLM are plotted on the scale of the linear predictor, but the vertical axis is labelled on the response scale. This preserves the linear structure of the model while permitting interpretation on what is usually a more familiar scale. This approach may also be used with linear models, for example to display effects on the scale of the response even if the data are analyzed on a transformed scale, such as log or square-root.

In a polytomous (multinomial or proportional-odds) logit model, by default effects are plotted on the probability scale; they may be alternatively plotted on the scale of the individual-level logits.

Value

The summary method for "eff" objects returns a "summary.eff" object with the following components (those pertaining to confidence limits need not be present):

header	a character string to label the effect.
effect	an array containing the estimated effect.
lower.header	a character string to label the lower confidence limits.
lower	an array containing the lower confidence limits.
upper.header	a character string to label the upper confidence limits.
upper	an array containing the upper confidence limits.

The [method for "efflist" objects is used to subset an "efflist" object and returns an object of the same class.

Author(s)

John Fox <jfox@mcmaster.ca> and Jangman Hong.

See Also

effect, allEffects, xyplot, densityplot, print.trellis rainbow_hcl, sequential_hcl

Examples

```
# also see examples in ?effect

mod.cowles <- glm(volunteer ~ sex + neuroticism*extraversion,
    data=Cowles, family=binomial)
eff.cowles <- allEffects(mod.cowles, xlevels=list(extraversion=seq(0, 24, 6)))
eff.cowles
as.data.frame(eff.cowles[[2]]) # neuroticism*extraversion interaction

plot(eff.cowles, 'sex', ylab="Prob(Volunteer)", grid=TRUE, rotx=90)

plot(eff.cowles, 'neuroticism:extraversion', ylab="Prob(Volunteer)",
    ticks=list(at=c(.1,.25,.5,.75,.9)))

## Not run:
# change color of the confidence bands to 'black' with .15 transparency
plot(eff.cowles, 'neuroticism:extraversion', ylab="Prob(Volunteer)",
    ticks=list(at=c(.1,.25,.5,.75,.9)), band.colors="red", band.transparency=.3)

plot(eff.cowles, 'neuroticism:extraversion', multiline=TRUE,
    ylab="Prob(Volunteer)", key.args = list(x = 0.75, y = 0.75, corner = c(0, 0)))

# use probability scale in place of logit scale, all lines are black.
```

```
plot(eff.cowles, 'neuroticism:extraversion', multiline=TRUE,
    ylab="Prob(Volunteer)", key.args = list(x = 0.75, y = 0.75, corner = c(0, 0)),
    colors="black", lines=1:8,
    ci.style="bands", rescale.axis=FALSE, band.colors=palette())

plot(effect('sex:neuroticism:extraversion', mod.cowles,
    xlevels=list(extraversion=seq(0, 24, 6))), multiline=TRUE)

plot(effect('sex:neuroticism:extraversion', mod.cowles,
    xlevels=list(extraversion=seq(0, 24, 6))), multiline=TRUE,
    rescale.axis=FALSE, ci.style="bands")

## End(Not run)

if (require(nnet)){
mod.beps <- multinom(vote ~ age + gender + economic.cond.national +
economic.cond.household + Blair + Hague + Kennedy +
Europe*political.knowledge, data=BEPS)
## Not run:
plot(effect("Europe*political.knowledge", mod.beps,
xlevels=list(political.knowledge=0:3)))

## End(Not run)

plot(effect("Europe*political.knowledge", mod.beps,
xlevels=list(political.knowledge=0:3),
given.values=c(gendermale=0.5)),
style="stacked", colors=c("blue", "red", "orange"), rug=FALSE)
}

if (require(MASS)){
mod.wvs <- polr(poverty ~ gender + religion + degree + country*poly(age,3),
data=WVS)
plot(effect("country*poly(age, 3)", mod.wvs))

## Not run:
plot(effect("country*poly(age, 3)", mod.wvs), style="stacked",
colors=c("gray75", "gray50", "gray25"))

plot(effect("country*poly(age, 3)", latent=TRUE, mod.wvs))

## End(Not run)
}

mod.pres <- lm(prestige ~ log(income, 10) + poly(education, 3) + poly(women, 2),
    data=Prestige)
eff.pres <- allEffects(mod.pres, default.levels=50)
## Not run:
plot(eff.pres)
plot(eff.pres[1:2])

## End(Not run)
plot(eff.pres[1],
```

```
transform.x=list(income=list(trans=log10, inverse=function(x) 10^x)),
ticks.x=list(income=list(at=c(1000, 2000, 5000, 10000, 20000))))
```

`TitanicSurvival` *Survival of Passengers on the Titanic*

Description

Information on the survival status, sex, age, and passenger class of 1309 passengers in the Titanic disaster of 1912.

Usage

```
TitanicSurvival
```

Format

A data frame with 1309 observations on the following 4 variables.

`survived` `no` or `yes`.

`sex` `female` or `male`

`age` in years (and for some children, fractions of a year); age is missing for 263 of the passengers.

`passengerClass` `1st`, `2nd`, or `3rd` class.

Details

This is part of a larger data set compiled by Thomas Cason. Many additional details are given in the sources cited below.

Source

Data set `titanic3` from http://biostat.mc.vanderbilt.edu/twiki/bin/view/Main/DataSets.

References

http://www.encyclopedia-titanica.org/

F. E. Harrell, Jr. (2001) *Regression Modeling Strategies* New York: Springer.

Examples

```
summary(TitanicSurvival)

titanic <- glm(survived ~ (passengerClass + sex + age)^2,
               data=TitanicSurvival, family=binomial)

titanic.all <- allEffects(titanic, typical=median,
given.values=c(passengerClass2nd=1/3, passengerClass3rd=1/3, sexmale=0.5))
```

```
plot(titanic.all, ticks=list(at=c(.01, .05, seq(.1, .9, by=.2), .95, .99)),
    ask=FALSE)

plot(effect("passengerClass*sex*age", titanic, xlevels=list(age=0:65)),
ticks=list(at=c(.001, .005, .01, .05, seq(.1, .9, by=.2), .95, .99, .995)))
```

`Wells` *Well Switching in Bangladesh*

Description

Data on whether or not households in Bangladesh changed the wells that they were using.

Usage

```
Wells
```

Format

A data frame with 3020 observations on the following 5 variables.

`switch` whether or not the household switched to another well from an unsafe well: `no` or `yes`.

`arsenic` the level of arsenic contamination in the household's original well, in hundreds of micrograms per liter; all are above 0.5, which was the level identified as "safe".

`distance` in meters to the closest known safe well.

`education` in years of the head of the household.

`association` whether or not any members of the household participated in any community organizations: `no` or `yes`.

Details

The data are for an area of Arahazar upazila, Bangladesh. The researchers labelled each well with its level of arsenic and an indication of whether the well was "safe" or "unsafe." Those using unsafe wells were encouraged to switch. After several years, it was determined whether each household using an unsafe well had changed its well. These data are used by Gelman and Hill (2007) for a logistic-regression example.

Source

http://www.stat.columbia.edu/~gelman/arm/examples/arsenic/wells.dat.

References

A. Gelman and J. Hill (2007) *Data Analysis Using Regression and Multilevel/Hierarchical Models.* Cambridge: Cambridge University Press.

Examples

```
summary(Wells)
```

WVS *World Values Surveys*

Description

Data from the World Values Surveys 1995-1997 for Australia, Norway, Sweden, and the United States.

Usage

```
WVS
```

Format

A data frame with 5381 observations on the following 6 variables.

`poverty` "Do you think that what the government is doing for people in poverty in this country is about the right amount, too much, or too little?" (ordered): `Too Little`, `About Right`, `Too Much`.

`religion` Member of a religion: `no` or `yes`.

`degree` Held a university degree: `no` or `yes`.

`country` `Australia`, `Norway`, `Sweden`, or `USA`.

`age` in years.

`gender` `male` or `female`.

References

J. Fox and R. Andersen (2006) Effect displays for multinomial and proportional-odds logit models. *Sociological Methodology* **36**, 225–255.

Examples

```
summary(WVS)


require(splines)  # for bs()
wvs <- polr(poverty ~ gender + country*(religion + degree + bs(age, 4)), data=WVS)

plot(effect("country*bs(age,4)", wvs, xlevels=list(age=18:83),
  given.values=c(gendermale=0.5)), rug=FALSE)
plot(effect("country*bs(age,4)", wvs, xlevels=list(age=18:83),
  given.values=c(gendermale=0.5)), rug=FALSE, style="stacked")
plot(effect("country*bs(age,4)", wvs, xlevels=list(age=18:83),
  given.values=c(gendermale=0.5), latent=TRUE), rug=FALSE)
```

Index