# Package 'visreg'

August 26, 2014

**Title** Visualization of regression models

**Version** 2.0-6

**Date** 2014-08-26

**Author** Patrick Breheny, Woodrow Burchett

**Maintainer** Patrick Breheny <patrick-breheny@uiowa.edu>

**Imports** lattice

**Suggests** rgl, survival, MASS, nlme

**Description** Provides convenient interface for constructing plots to visualize the fit of regression models arising from a wide variety of models in R (lm, glm, coxph, rlm, gam, locfit, etc.)

**License** GPL-2

**URL** http://myweb.uiowa.edu/pbreheny/publications/visreg.pdf

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2014-08-26 20:17:46

## R topics documented:

## visreg-package *Visualization of regression models*

### Description

visreg provides a number of plotting functions for visualizing fitted regression models: regression functions, confidence bands, partial residuals, interactions, and more.

### Details

This package allows the use of visreg and visreg2d, functions for visualizing regression models. See example below for the most basic use, and the help pages for each function for details. Also see the cited manuscript for additional details. If you have a question or feature request, please e-mail me at <patrick-breheny@uiowa.edu>.

### Author(s)

Patrick Breheny and Woodrow Burchett

Maintainer: Patrick Breheny <patrick-breheny@uiowa.edu>

### References

Breheny, P. and Burchett, W. (2013), Visualizing regression models using visreg. http://myweb.uiowa.edu/pbreheny/publications/visreg.pdf

### See Also

visreg visreg2d visreg-faq

### Examples

```
fit <- lm(Ozone ~ Solar.R + Wind + Temp, data=airquality)
visreg(fit, "Wind")
```

## visreg *Visualization of regression functions for a single variable*

### Description

A function for visualizing regression models quickly and easily. Default plots contain a confidence band, prediction line, and partial residuals. Factors, transformations, conditioning, interactions, and a variety of other options are supported.

## Usage

```
visreg(fit, xvar, by, overlay=FALSE, breaks=3, type=c("conditional",
"contrast", "effect"), trans=I, scale=c("linear","response"), xtrans,
alpha=.05, nn=101, cond=list(), print.cond=missing(by) &
(max(attr(terms(formula(fit)), "order"))>1), whitespace=0.2,
partial=identical(trans, I), band=TRUE, rug=!partial, jitter=FALSE,
strip.names=is.numeric(attr(v, "lev")), legend=TRUE, ask=TRUE,
line.par=NULL, fill.par=NULL, points.par=NULL, ...)
```

## Arguments

| | |
|---|---|
| `fit` | The fitted model object you wish to visualize. Any object with 'predict' and 'model.frame' methods are supported, including lm, glm, gam, rlm, coxph, and many more. |
| `xvar` | Character string specifying the variable to be put on the x-axis of your plot. Both continuous variables and factors are supported. |
| `by` | (Optional) A variable allowing you to divide your plot into cross-sections based on levels of the by variable; particularly useful for visualizing models with interactions. Supplied as a character string. Uses the lattice package. Both continuous variables and factors are supported. |
| `overlay` | When by is specified, by default separate panels are used to display each cross-section. If `overlay=TRUE`, these cross-sections are overlaid on top of each other in a single plot. |
| `breaks` | If a continuous variable is used for the by option, the `breaks` argument controls the values at which the cross-sections are taken. By default, cross-sections are taken at the 10th, 50th, and 90th quantiles. If `breaks` is a single number, it specifies the number of breaks. If `breaks` is a vector of numbers, it specifies the values at which the cross-sections are to be taken. Each partial residuals appears exactly once in the plot, in the panel it is closest to. |
| `type` | The type of plot to be produced. The following options are supported:<br>• If 'conditional' is selected, the plot returned shows the value of the variable on the x-axis and the change in response on the y-axis, holding all other variables constant (by default, median for numeric variables and most common category for factors).<br>• If 'contrast' is selected, the plot returned shows the effect on the expected value of the response by moving the x variable away from a reference point on the x-axis (for numeric variables, this is taken to be the mean).<br>For more details, see references. |
| `trans` | (Optional) A function specifying a transformation for the vertical axis. |
| `scale` | By default, the model is plotted on the scale of the linear predictor. If `scale='response'` for a glm, the inverse link function will be applied so that the model is plotted on the scale of the original response. |
| `xtrans` | (Optional) A function specifying a transformation for the horizontal axis. Note that, for model terms such as `log(x)`, visreg automatically plots on the original axis (see examples). |

| | |
|---|---|
| alpha | Alpha level (1-coverage) for the confidence band displayed in the plot (default: 0.05). |
| nn | Controls the smoothness of the line and confidence band. Increasing this number will add to the computational burden, but produce a smoother plot (default: 101). |
| cond | Named list specifying conditional values of other explanatory variables. By default, conditional plots in visreg are constructed by filling in other explanatory variables with the median (for numeric variables) or most common category (for factors), but this can be overridden by specifying their values using cond (see examples). |
| print.cond | If print.cond=TRUE, the explanatory variable values conditioned on in a conditional plot are printed to the console (default: FALSE). If print.cond=TRUE and type="contrast", the conditions will still be printed, but they have no bearing on the plot unless interactions are present. |
| whitespace | When xvar is a factor, whitespace determines the amount of space in between factors on the x-axis. Default is 0.2, meaning that 20 percent of the horizontal axis is whitespace. |
| partial | If partial=TRUE (the default unless a transformation has been applied), partial residuals are shown on the plot. |
| band | If band=TRUE (the default), confidence bands are shown on the plot. |
| rug | A rug is an alternative to partial residuals, showing the location on the horizontal axis corresponding to observations, but not displaying any information along the vertical axis. By defaut, if no transformation has been applied, partial residuals are plotted and a rug is not; these defaults are reversed if the vertical axis has been transformed. If one wants both to be plotted, both rug=TRUE and partial=TRUE need to be specified. Two types of rug plots are available. If rug=1 or rug=TRUE, then a basic rug is drawn on the bottom. If rug=2, then separate rugs are drawn on the top for observations with positive residuals and on the bottom for observations with negative residuals. Such plots are particularly useful in logistic regression (see examples). |
| jitter | Adds a small amount of noise to xvar. Potentially useful if many observations have exactly the same value. Default is FALSE. |
| strip.names | When by=TRUE, strip.names=TRUE adds the name of the by variable to the strip at the top of each panel. Default is FALSE for factors and TRUE for numeric by variables. strip.names can also be a character vector, in which case it replaces the strip names altogether with values chosen by the user. |
| legend | For overlay plots, (overlay=TRUE), should visreg create a legend? If legend=TRUE (the default), a legend is placed in the top margin. |
| ask | By default (ask=TRUE), if multiple plots are requested (for example, by passing a vector to xvar) and the result would cause some of the graphs to be on one figure and others on a different figure, visreg will prompt the user before moving to the next figure. The prompting can be turned off with ask=FALSE. |
| line.par | List of parameters (see par) to pass to lines(...) when lines are drawn in the plots. |
| fill.par | List of parameters (see par) to pass to polygon(...) when shaded confidence regions are drawn in the plots. |

| | |
|---|---|
| points.par | List of parameters (see par) to pass to points(...) when partial residuals are drawn in the plots. |
| ... | Graphical parameters can be passed to the function to customize the plots. If by=TRUE, lattice parameters can be passed, such as layout (see examples below). |

## Value

In addition to providing plots, the visreg function also invisibly returns the data frames, estimates, confidence intervals, and residuals used in the creation of its plots (see last example).

## Author(s)

Patrick Breheny and Woodrow Burchett

## References

Breheny, P. and Burchett, W. (2012), Visualizing regression models using visreg. http://myweb.uiowa.edu/pbreheny/publications/visreg.pdf

## See Also

visreg2d visreg-faq

## Examples

```
###################
## Linear models ##
###################

## Basic
fit <- lm(Ozone ~ Solar.R + Wind + Temp,data=airquality)
visreg(fit)
visreg(fit, "Wind", type="contrast")
visreg(fit, "Wind", type="conditional")
visreg(fit, "Wind", type="conditional", rug=TRUE)
visreg(fit,c("Solar.R","Wind","Temp"))

## Factors
airquality$Heat <- cut(airquality$Temp,3,labels=c("Cool","Mild","Hot"))
fit.heat <- lm(Ozone ~ Solar.R + Wind + Heat,data=airquality)
visreg(fit.heat,"Heat",type="contrast")
visreg(fit.heat,"Heat",type="conditional")
## Use of whitespace option
visreg(fit.heat,"Heat",whitespace=.1)
visreg(fit.heat,"Heat",whitespace=.5)

## Transformations
fit1 <- lm(Ozone ~ Solar.R + Wind + Temp + I(Wind^2),data=airquality)
fit2 <- lm(log(Ozone) ~ Solar.R + Wind + Temp,data=airquality)
fit3 <- lm(log(Ozone) ~ Solar.R + Wind + Temp +
  I(Wind^2),data=airquality)
```

```
visreg(fit1,"Wind")
visreg(fit2,"Wind",trans=exp,ylab="Ozone")
visreg(fit3,"Wind",trans=exp,ylab="Ozone")

## Conditioning
visreg(fit, "Wind", cond=list(Temp=50))
visreg(fit, "Wind", print.cond=TRUE)
visreg(fit, "Wind", cond=list(Temp=100))

## Interactions
fit.in1 <- lm(Ozone~ Solar.R + Wind*Heat,data=airquality)
visreg(fit.in1, "Wind", by="Heat")
visreg(fit.in1, "Heat", by="Wind")
visreg(fit.in1, "Wind", by="Heat", type="contrast")
visreg(fit.in1, "Wind", cond=list(Heat="Cool"), type="contrast")
visreg(fit.in1, "Wind", cond=list(Heat="Hot"), type="contrast")
visreg(fit.in1, "Heat", by="Wind", breaks=6)
visreg(fit.in1, "Heat", by="Wind", breaks=c(0,10,20))

## Overlay
visreg(fit.in1, "Wind", by="Heat", overlay=TRUE)
visreg(fit.in1, "Wind", by="Heat", overlay=TRUE, partial=FALSE)
visreg(fit.in1, "Wind", by="Heat", overlay=TRUE, partial=FALSE, band=FALSE)
visreg(fit.in1, "Wind", by="Heat", overlay=TRUE, partial=FALSE,
band=FALSE, type="contrast")

## Changing appearance
visreg(fit, "Wind", line=list(col="blue"), points=list(cex=1, pch=1))

######################
## Nonlinear models ##
######################

## Logistic regression
data("birthwt",package="MASS")
birthwt$race <- factor(birthwt$race,labels=c("White","Black","Other"))
birthwt$smoke <- factor(birthwt$smoke,labels=c("Nonsmoker","Smoker"))
fit <- glm(low~age+race+smoke+lwt,data=birthwt,family="binomial")
visreg(fit,"lwt",xlab="Mother's Weight",
  ylab="Log odds (low birthweight)")
visreg(fit, "lwt", scale="response", partial=FALSE, xlab="Mother's
Weight", ylab="P(low birthweight)")
visreg(fit, "lwt", scale="response", partial=FALSE, xlab="Mother's
Weight", ylab="P(low birthweight)", rug=2)

## Proportional hazards
require(survival)
data(ovarian)
ovarian$rx <- factor(ovarian$rx)
fit <- coxph(Surv(futime,fustat)~age+rx,data=ovarian)
visreg(fit,"age",ylab="log(Hazard ratio)")

## Robust regression
```

```
require(MASS)
fit <- rlm(Ozone ~ Solar.R + Wind*Heat,data=airquality)
visreg(fit,"Wind",cond=list(Heat="Mild"))

## And more...; anything with a 'predict' method should work

## Return raw components of plot
v <- visreg(fit,"Wind",cond=list(Heat="Mild"))
```

---

visreg-faq *Frequently Asked Questions for visreg*

---

## Description

This page tries to answer some of the questions that I get asked most often about how to use the visreg package. If you have a question that does not appear below, I can be reached at <patrick-breheny@uiowa.edu>.

## Frequent asked questions

1. **What is the difference between 'conditional' and 'contrast' plots?**
   Suppose our data looked like:

| SBP | Sex | Age |
|---|---|---|
| 140 | M | 56 |
| 135 | F | 47 |
| ... | | |

   we fit a model with

   `fit <- lm(SBP~Sex+Age)`

   and we want to plot the relationship between Age and SBP. A 'conditional' plot illustrates the relationship between the two, conditional on the sex being, say, Male (the default in visreg is to choose the most common category).

   The 'contrast' plot in visreg, on the other hand, illustrates the effect on SBP of a *change* in age – the default in visreg is to use the mean age as the reference point for this change. Since the above model does not have an interaction, this effect will be the same for men and women, and thus does not require you to specify a sex for the plot.

   Both conditional and contrast plots answer subtly different questions, and both are useful in different situations.

2. **Can visreg can be used for mixed models (i.e., from the 'nlme' or 'lme4' packages)?**
   Sort of. The underlying basis on which visreg operates is by using the predict method to plot predictions from the model. Predictions for mixed models are complicated. In particular, there is no `se.fit` option provided by the `predict` methods in the `nlme` and `lme4` packages, so you cannot obtain confidence bands for conditional plots. Nevertheless, `visreg` will produce plots of estimated coefficients and partial residuals.

Keep in mind, that depending on what sort of predictions (BLUPs) you are interested in, you may need to manually control the inclusion of random effects in your predictions. By default, visreg includes no random effects (i.e., `level=0` for `nlme` models and `REform=NA` for `lme4` models). If you are including a random effect as a `by` variable in `visreg`, you most likely want to add those effects back in, and you will have to do so manually, by directly specifying the appropriate `level` or `REform` argument to `predict` (see `?predict.nlme` or `?predict.merMod`). Handling this appropriately is the user's responsibility; I cannot hope to automatically decide this for all possible mixed models that could be passed to visreg.

As mentioned above, you cannot obtain confidence bands for conditional plots. In the words of the authors of the `lme4` package, "There is no option for computing standard errors of predictions because it is difficult to define an efficient method that incorporates uncertainty in the variance parameters"; hence no `se.fit` option. You can, however, get confidence bands for `'contrast'` plots. In a contrast plot, the random effects cancel and the above issue is avoided.

If you are running into difficulty using `visreg` with mixed models, feel free to e-mail me; mixed models have been less extensively tested with `visreg` than fixed-effect models, and there may still be bugs to work out.

3. **How do I use visreg for a model with offset terms?**

   By default, visreg is set up to provide conditional plots in which all other terms are set to their median value (or most common category). This includes offset terms. It is not uncommon, however, to want to see results with the offset included. To obtain these results, one needs to specify the offset among the arguments to cond. For example, using the Insurance data from the MASS package:

   ```
   utils::data(Insurance, package="MASS")
   fit <- glm(Claims ~ District + Group + Age +
   offset(log(Holders)), data = Insurance, family = poisson)
   visreg(fit, "Group", scale="response")
   ```

   This will provide the model's predictions for the expected number of claims given the median number of holders (here, 136). To obtain the expected number of claims per holder, we need to specify Holders=1 in cond:

   ```
   visreg(fit, "Group", scale="response", cond=list(Holders=1))
   ```

   Note also that to ensure proper functionality with all of visreg's options, the use of the `offset()` function, rather than the `offset=` argument, is recommended.

## Author(s)

Patrick Breheny and Woodrow Burchett

Maintainer: Patrick Breheny <patrick-breheny@uiowa.edu>

## References

Breheny, P. and Burchett, W. (2013), Visualizing regression models using visreg. http://myweb.uiowa.edu/pbreheny/publications/visreg.pdf

visreg2d *Visualization of regression functions for two variables*

## Description

A function used to visualize how two variables interact to affect the response in regression models.

## Usage

```
visreg2d(fit, xvar, yvar, type=c("conditional", "contrast", "effect"),
trans=I, scale=c("linear","response"),
plot.type=c("image","persp","rgl"), nn=ifelse(plot.type=="persp",49,99),
cond=list(), print.cond=FALSE, whitespace=0.2, ...)
```

## Arguments

| | |
|---|---|
| fit | The fitted model object you wish to visualize. Any object with 'predict' and 'model.frame' methods are supported, including lm, glm, gam, rlm, coxph, and many more. |
| xvar | Character string specifying the variable to be put on the x-axis of your plot. Both continuous variables and factors are supported. |
| yvar | Character string specifying the variable to be put on the y-axis of your plot. Both continuous variables and factors are supported. |
| type | The type of plot to be produced. The following options are supported:<br>• If 'conditional' is selected, the plot returned shows the value of the variable on the x-axis and the change in response on the y-axis, holding all other variables constant (by default, median for numeric variables and most common category for factors).<br>• If 'contrast' is selected, the plot returned shows the effect on the expected value of the response by moving the x variable away from a reference point on the x-axis (for numeric variables, this is taken to be the mean).<br>For more details, see references. |
| trans | (Optional) A function specifying a transformation for the vertical axis. |
| scale | By default, the model is plotted on the scale of the linear predictor. If `scale='response'` for a glm, the inverse link function will be applied so that the model is plotted on the scale of the original response. |
| plot.type | The style of plot to be produced. The following three options are supported:<br>• `'image'`, a filled contour plot.<br>• `'persp'`, a 3 dimensional perspective plot.<br>• `'rgl'`, a version of the perspective plot that can be rotated. Note: requires the rgl package to use. |
| nn | Resolution of the three dimensional plot. Higher values will results in a smoother looking plot. |

| | |
|---|---|
| cond | Named list specifying conditional values of other explanatory variables. By default, conditional plots in visreg are constructed by filling in other explanatory variables with the median (for numeric variables) or most common category (for factors), but this can be overridden by specifying their values using cond (see examples). |
| print.cond | If print.cond==TRUE, the explanatory variable values conditioned on in a conditional plot are printed to the console (default: FALSE). If print.cond==TRUE and type=="contrast", the conditions will still be printed, but they have no bearing on the plot unless interactions are present. |
| whitespace | When xvar or yvar is a factor, whitespace determines the amount of space in between the factors. Default is 0.2, meaning that 20 percent of the axis is whitespace. |
| ... | Graphical parameters can be passed to the function to customize the plots. |

## Value

In addition to providing plots, the visreg2d function also invisibly returns the x, y, and Z values used in the creation of its plots.

## Author(s)

Patrick Breheny and Woodrow Burchett

## References

Breheny, P. and Burchett, W. (2012), Visualizing regression models using visreg. http://myweb.uiowa.edu/pbreheny/publications/visreg.pdf

## See Also

visreg

## Examples

```
fit <- lm(Ozone ~ Solar.R + Wind + Temp + I(Wind^2) + I(Temp^2) +
I(Wind*Temp)+I(Wind*Temp^2) + I(Temp*Wind^2) + I(Temp^2*Wind^2),
data=airquality)

visreg2d(fit,x="Wind",y="Temp",plot.type="image")
visreg2d(fit,x="Wind",y="Temp",plot.type="persp")

## Requires the rgl package
## Not run:
visreg2d(fit,x="Wind",y="Temp",plot.type="rgl")

## End(Not run)
```

# Index